

AFIT/DS/ENG/98-13

Automatic Target Cueing
of Hyperspectral Image Data

DISSERTATION
Terry A. Wilson
Captain, USAF

AFIT/DS/ENG/98-13

19980924 041

DTIC QUALITY INSPECTED 1

Approved for public release; distribution unlimited

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

AFIT/DS/ENG/98-13

Automatic Target Cueing
of Hyperspectral Image Data

DISSERTATION

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

Terry A. Wilson, B.S.E.E., M.S.E.E.

Captain, USAF

September, 1998


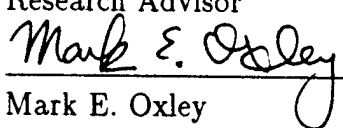
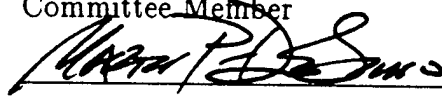
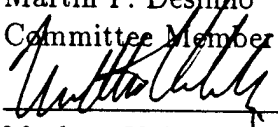

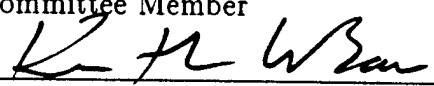
Approved for public release; distribution unlimited

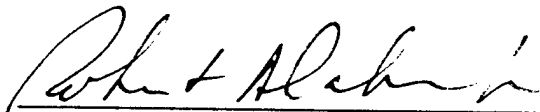
Automatic Target Cueing
of Hyperspectral Image Data

Terry A. Wilson, B.S.E.E., M.S.E.E.

Captain, USAF

Approved:

	<u>10 Aug 98</u>
Steven K. Rogers Research Advisor	Date
	<u>10 Aug 98</u>
Mark E. Oxley Committee Member	Date
	<u>Aug 10, 1998</u>
Martin P. Desimio Committee Member	Date
	<u>10 Aug 1998</u>
Mathew Kabrisky Committee Member	Date
	<u>10 Aug 98</u>
Thomas Rathbun Committee Member	Date
	<u>10 AUG 98</u>
Kenneth Bauer Jr. Dean's Representative	Date


Robert A. Calico, Jr
Dean

Acknowledgements

I have found in life that nothing of import is accomplished without the support of many people. This has certainly held true for my education at AFIT. I would like to take this opportunity to say thank you to some of the many people who made my work at AFIT not only an academic learning experience, but a personal and spiritual growing experience as well.

First, I am thankful to have had a great advisor, Dr. Steven Rogers. His dedication to excellence was an inspiration. I also appreciate the fact that he always found the time to meet with me even though he had more on his schedule than two people. I would also like to thank my committee members (Dr. Oxley, Dr. Kabrisky, Dr. DeSimio, and Dr. Rathbun) for their own unique talents and abilities. It seemed that no matter the problem, I could find at least one of them that could help. Their multiple talents also made me answer questions and address issues that considerably broadened my own education. I would like to extend a special thanks to Dr. Oxley for all of the mathematical discussions we had. He seemed to never tire of answering questions or listening to ideas that I had. Thank you also to Mr. Dan Zambon and Mr. Dave Doak for keeping the computers running and for responding in a very quick manner when anything did go wrong.

On a personal note I would like to thank John Keller for his friendship throughout this challenging PhD program. His sense of humor was a constant source of enjoyment and at times a stress relief. Finally, and most of all, I want to thank my wife Deborah and my children Daniel and Tyler for their support and sacrifices throughout this long process. They are the ones that give true meaning to my life.

Terry A. Wilson

Table of Contents

	Page
Acknowledgements	iv
List of Figures	viii
List of Tables	x
Abstract	xi
 I. Introduction	 1-1
1.1 Motivational Background	1-1
1.2 Problem Statement	1-3
1.3 Scope	1-3
1.4 Dissertation Organization	1-5
 II. Background	 2-1
2.1 Hierarchical Image Fusion Techniques	2-1
2.2 Wavelet Multiresolution Decomposition and Reconstruction . .	2-6
2.3 Artificial Neural Network Architectures	2-9
2.3.1 Fixed Architecture Feed-Forward ANNs	2-11
2.3.2 Adaptive Architecture Feed-Forward ANNs	2-25
2.4 Summary	2-28
 III. Wavelet-Based Hierarchical Image Fusion with Contrast Sensitivity Saliency	 3-1
3.1 Background	3-1
3.2 Fusion Algorithm	3-2
3.2.1 Stage I: Wavelet Decomposition	3-2
3.2.2 Stage II: Fusion	3-3
3.2.3 Stage III: Wavelet Reconstruction	3-9

	Page
3.3 Experiments	3-9
3.4 Results	3-10
3.5 Fusion of AVIRIS Hyperspectral Image Data	3-19
3.5.1 Fusion of IR and Visible Bands of AVIRIS Image Data	3-19
3.5.2 Fusion of Sets of Hyperspectral AVIRIS Image Data . .	3-24
3.6 Conclusions	3-24
IV. Radial Basis Function Iterative Construction Algorithm (RICA)	4-1
4.1 Selecting and Training The Hidden Layer Nodes	4-1
4.1.1 Algorithm for Selecting and Training The HLN's	4-4
4.1.2 Shapiro-Wilk Goodness-of-Fit Algorithm	4-5
4.1.3 Mahalanobis Distance Clustering Algorithm	4-6
4.2 Training The Output Layer	4-8
4.2.1 Ho-Kashyap Algorithm	4-9
4.3 RICA: a minimum MSE approximation to a Bayes optimal dis-	
criminant function	4-11
4.4 Experiments	4-16
4.5 Results	4-19
4.6 Application to Automatic Target Cueing of Hyperspectral Image	
Data	4-20
4.6.1 Motivation	4-21
4.6.2 Experiments and Results	4-22
4.7 Conclusions	4-29
V. Conclusions and Contributions	5-1
5.1 Conclusions	5-1
5.2 Contributions	5-2
5.3 Recommendations for Future Work	5-3
Appendix A. Additional Test Results	A-1

	Page
Bibliography	BIB-1
Vita	VITA-1

List of Figures

Figure		Page
2.1.	Multi-resolution Pyramid Algorithm	2-3
2.2.	Reconstruction Scale and Wavelet Functions	2-9
2.3.	Decomposition Scale and Wavelet Functions	2-10
2.4.	Wavelet frequency responses.	2-10
2.5.	Decomposition flow chart.	2-11
2.6.	Example of Level 1 of a Discrete Wavelet Decomposition	2-12
2.7.	Example of Level 2 of a Discrete Wavelet Decomposition	2-13
2.8.	Reconstruction flow chart.	2-14
2.9.	Typical Feed-forward ANN Architecture [83].	2-14
2.10.	Example of Kohonen SOFM Using a single 2D Gaussian Example . . .	2-18
2.11.	Example of Kohonen SOFM Using two 2D Gaussians	2-18
2.12.	Example of Kohonen SOFM Using four 2D Gaussians	2-19
3.1.	Fusion Pyramid Algorithm	3-3
3.2.	Contrast sensitivity weight matrix C	3-6
3.3.	Diagram of How The Distance From CRT Relates to Spatial Frequencies	3-7
3.4.	Images Only Differ by Some Constant Bias	3-8
3.5.	Hyperspectral Image Cube Representation From an AVIRIS Sensor . .	3-10
3.6.	Three Test Images of Lenna with 5db energy SNR of Uncorrelated Noise Added	3-11
3.7.	Three Test Images of Lenna with 5db energy SNR of Correlated Noise Added	3-12
3.8.	Fusion Results of Test Images Using Uncorrelated Noise Added to Lenna.	3-15
3.9.	Fusion Results of Test Images Using Correlated Noise Added to Lenna.	3-16
3.10.	Fusion Results of Test Images Using Correlated Noise Added to Band30.	3-17
3.11.	Original Image of Lenna and a Composite Image Showing Uncorrelated Noise Added.	3-19

Figure		Page
3.12.	Original Image of Lenna and a Composite Image Showing Correlated Noise Added.	3-20
3.13.	Original Image of Band 30 and a Composite Image Showing Correlated Noise Added.	3-20
3.14.	AVIRIS Images Representing Bands 30, 60, and 90.	3-22
3.15.	Fusion Results AVIRIS Bands 30,60, and 90.	3-23
3.16.	Wavelet-Based Contrast Sensitivity Fusion Results of AVIRIS Bands 30 to 40	3-24
4.1.	This Figure illustrates the process involved in splitting the data set into two parts and measuring the Shapiro-Wilk test on the original cluster and then the two new clusters.	4-4
4.2.	Plot Of Two Dimensions of The Benchmark Test Data Set.	4-18
4.3.	Plot of 3 class sample data.	4-20

List of Tables

Table		Page
3.1.	Signal-to-Noise Ratios for the Lenna Modified Test Images.	3-13
3.2.	Signal-to-Noise Ratios for the Band 30 Modified Test Images.	3-13
3.3.	Signal-to-Noise Ratios for the Fused Test Images Containing Uncorrelated Noise.	3-14
3.4.	Signal-to-Noise Ratios for the Fused Test Images Containing Correlated Noise.	3-14
4.1.	Results of 24 trials using an Optical Character Recognition feature data set.	4-20
4.2.	Results of Using Bench Data with a known Bayes Optimal of a 91.0 percent accuracy rate.	4-21
4.3.	Results of 30 trials using the Non-fused and Fused data sets.	4-23
4.4.	ATC Results Using 21 Hyperspectral Images.	4-29
A.1.	Results of 30 trials using 3 class sample test data.	A-1
A.2.	Results of 24 trials using Infrared Target imagery.	A-1

Abstract

Modern imaging sensors produce vast amounts of data, overwhelming human analysts. One such sensor is the Airborne Visible and Infrared Imaging Spectrometer (AVIRIS) hyperspectral sensor. The AVIRIS sensor simultaneously collects data in 224 spectral bands that range from $0.4\mu\text{m}$ to $2.5\mu\text{m}$ in approximately 10nm increments, producing 224 images, each representing a single spectral band. Autonomous systems are required that can fuse “important” spectral bands and then classify regions of interest if all of this data is to be exploited. This dissertation presents a comprehensive solution that consists of a new physiologically motivated fusion algorithm and a novel Bayes optimal self-architecting classifier that processes the outputs of the fusion algorithm. The fusion algorithm which uses a contrast sensitivity weighted wavelet-based multiresolution analysis is shown to outperform other fusion algorithms in both visual aesthetics and signal-to-noise ratios. The self-architecting classifier is a Radial Basis Function (RBF) Iterative Construction Algorithm (RICA) that is designed to autonomously determine the size of its network architecture for optimal classification performance. RICA is shown to outperform several neural network algorithms, including a fixed architecture multi-layer Perceptron (MLP), a fixed architecture RBF, and an adaptive architecture MLP. A proof is also presented demonstrating that RICA produces a network which is a minimum mean squared-error approximate to Bayes optimal discriminant functions. Finally, it is shown that this combination of image fusion and self-architecting classifier provide an excellent means to detect targets in hyperspectral sensor data.

Automatic Target Cueing of Hyperspectral Image Data

I. Introduction

1.1 Motivational Background

Over the last decade the Air Force has made drastic reductions in manpower. The focus now is on a smaller force that can be rapidly mobilized and deployed to any part of the world. The motto has become "more with less". In order to achieve "more with less" and to satisfy a national desire that American men and woman be removed from harm's way as much as possible, the Air Force is relying more and more on modern technology. One area of particular interest is Automatic Target Cueing (ATC) (i.e., using computers to process and analyze sensor image data and highlight regions of interest for possible targets). ATC can be used to reduce the burden of image analysts, pilots, or weapons officers who need to detect targets in image data. It can also be used as a front-end or preprocessor to an automated target recognition system. Two problems associated with ATC, which are addressed in this research, are sensor fusion and target detection.

Sensor fusion involves processing sensor data to provide an image analyst or target detection algorithm with an enhanced subset of the original data. Because the data can come from a single sensor or multiple sensors there is a need for image processing techniques that can fuse images from disparate sensors or multiple images produced by the same sensor [1, 2, 12, 93, 94]. For example, the Air Force needs to fuse information from multiple sensors to obtain a multi-spectral analysis of a given target area. The advantage of using multi-spectral data is that it provides better target detection and identification than a single wide-band sensor [12, 72] and it allows flexibility in choosing a particular narrow-spectral-band for individual types of targets. The disadvantage of using multiple sensors to obtain a multi-

spectral signature is that it is difficult and sometimes impossible to fully register (align in scale, rotation, and shift) the different input sources.

In an attempt to overcome the registration problem, while maintaining the multispectral information, research is being conducted using hyperspectral sensors that simultaneously collect data from hundreds of narrow spectral bands. However, there is a price to pay for the fully registered hyperspectral data.

The first is the cost of analyzing the image data generated by a hyperspectral sensor. Because hyperspectral sensors record information in hundreds of spectral bands and each band generates a full resolution single image, an image analyst has to read and analyze hundreds of images from a single pass of a hyperspectral sensor. Because human operators cannot physically or mentally integrate information from multiple source images [2, 93, 94], a method to fuse the relevant information into a single image, or at least a smaller subset of images, is needed.

A second cost results when an automated system is used to find targets in hyperspectral sensor data. These automated systems are inundated with the vast amounts of information hyperspectral sensors produce about a target area, including many irrelevant details. This increases the chances for false alarms and missed detections. Therefore, a fusion method that represents or preserves the details in the input images that are most relevant to the task at hand (i.e., target detection) and at the same time, provides better target detail [12], is needed.

While there are image fusion techniques which can be used to address some of the costs associated with using hyperspectral data, they are shown to favor noise in the input images or add additional noise based upon their decomposition and reconstruction methods [2, 12, 77, 93, 94, 96]. A sensor fusion technique is developed in this research that uses a combination of human visual perception and wavelet-based multiresolution analysis to fuse images from a hyperspectral sensor while overcoming the noise problems of the previous techniques.

After hyperspectral sensor fusion has occurred, the next step in ATC is to perform target detection using the enhanced imagery. One set of algorithms which may be useful in this role are Artificial Neural Networks (ANNs). ANNs have been successfully applied to

many types of classification problems, from breast cancer detection in digitized mammograms to target detection in infrared imagery [43,47,73,74,82]. One reason they are so successful is that they can be trained to correctly classify previously unseen test samples based upon the information in a set of training exemplars. The drawback is that most ANNs require human input to determine the size of the architecture needed to perform classification. There are ANNs that automatically determine their architectures, however they may not optimize the output for classification generalization or may only use *ad hoc* methods to select their architecture size [3, 8, 13, 15, 22–24, 27, 36, 37, 44, 45, 52, 54–56, 69, 70]. Steppe overcomes the “*ad hoc*” methods by introducing a statistically rigorous approach to MLP architecture selection [91], but her work requires multiple training sessions using various network architectures. While her method is statistically rigorous, it is computationally expensive. A unique self-architecting Radial Basis Function (RBF) ANN is developed that does not rely on *ad hoc* methods, optimizes its architecture for classification accuracy, and does not have the computational overhead associated with Steppe’s [91] and others’ adaptive approaches [22,68].

1.2 Problem Statement

The goal of this research is to develop both an image fusion algorithm and a self-architecting ANN to perform Automatic Target Cueing (ATC) of the resulting fused hyperspectral sensor data.

1.3 Scope

The scope of the image fusion portion of this research is to develop an image fusion method to address the problems facing hyperspectral sensors and to improve upon previous methods [2,12,77,93,94,96]. The best of those multiresolution methods had approximately 2-3 percent reconstruction error [12,96] based solely on decomposition and reconstruction. A wavelet-based approach, which has effectively zero reconstruction error, is designed and implemented in this research.

The design of the image fusion algorithm is based on the assumption that the images to be fused come from a single sensor that produces fully registered images, or from multiple sensors that have been pre-registered. It was also assumed that the data produced from the sensors may be treated as image data and that contrast sensitivity, as related to human visual perception, is the key for determining the salient features of an image [34, 61, 62, 79]. Furthermore, it was assumed that the key to multi-image fusion is to fuse based upon the salient features, (i.e., pattern primitives), and not at the pixel level alone [1, 2, 10–12, 93, 94].

Evaluation of the image fusion algorithms will be accomplished in two different ways. First, the algorithms' performances will be evaluated by fusing a set of test images with known image characteristics. The set of images will contain various levels of both uncorrelated and correlated random noise. Second, the algorithms' ability to fuse different spectral bands from the Airborne Visual and Infrared Imaging Spectrometer (AVIRIS) hyperspectral sensor will be demonstrated and visually evaluated.

After image fusion has been accomplished, an adaptive classifier is needed to perform classification of the fused data. The scope of the classification portion of this research is to design and implement an adaptive Radial Basis Function Iterative Construction Algorithm (RICA) that will adapt the size of its architecture to "fit" the underlying distribution of the training data and modify its parameters to optimize classification accuracy.

The performance of RICA will be compared with many other ANNs, both fixed and adaptive, using real world sample data sets. A proof is also presented which shows that RICA performs a minimum mean squared-error approximation to a Bayes optimal classifier. This proof also extends the proofs presented by Richard [71] and Ruck [81] by showing that no assumption about the underlying distribution of the data needs to be made.

Finally, an example of ATC using hyperspectral image data will be presented. The example is intended to show how the physiologically motivated fusion algorithm and RICA can work in concert to perform ATC of hyperspectral data. The ATC example will also demonstrate the information enhancement capability of the physiologically motivated fusion algorithm.

1.4 Dissertation Organization

This chapter presented the motivational background, the problem statement, and the scope of the work to be accomplished in this research. Chapter II provides background material on current fusion algorithms, ANNs, and their limitations. It also provides a brief discussion on wavelets. Chapter III presents the new physiologically motivated fusion algorithm and compares it with current techniques in the literature. Chapter IV describes the novel self-architecting classifier RICA, compares it to several ANNs, and presents an example of ATC using hyperspectral image data. It also details the proof showing that RICA performs a minimum mean squared-error approximation to a Bayes optimal classifier. Chapter V discusses conclusions, contributions, and recommendations.

II. Background

The goal of this research is to develop both an image fusion algorithm and an adaptive ANN to perform Automatic Target Cueing (ATC) of hyperspectral sensor data. The purpose of this chapter is to present background material on current hierarchical image fusion techniques and to discuss both fixed and adaptive (self-architecting) Artificial Neural Networks (ANNs) that may be used to process the outputs of the fused imagery.

2.1 Hierarchical Image Fusion Techniques

One of the goals of this research is to develop a fusion algorithm to fuse (combine) multiple bands of images from a hyperspectral sensor. The question is what type of fusion algorithm should be used. The most direct approach to image fusion is to simply sum and average the pixel values of the input images [12]. However, this can produce problems. Averaging images can cause features that were in one image but not in another to appear in the composite image at a much reduced contrast. Additionally, it can appear to be superimposed in the composite image, much like a double exposure in photography [12]. The desire is for the features in the composite image to maintain contrast and to appear "normal" (not a superimposed combination), especially if the fused image is to be viewed by a human analyst. Another method of fusion is to cut and paste features from the input images into a composite and then use some form of edge blurring to remove the false edges [93]. However, this can lead to edge artifacts that distort the fused image.

The problem with these techniques is that they all operate at the pixel level and not the feature level for image fusion. Burt introduced a pyramidal approach that was designed to perform fusion at the feature level and not the pixel level [12]. While his method overcame the problems of the more direct approaches discussed, his method introduced a new set of problems to overcome. Before Burt's and others' particular methods are discussed, a more general discussion of hierarchical image fusion techniques is presented.

Hierarchical image fusion techniques use a pyramidal approach to image fusion. Typically, they have three stages: 1) Decomposition, 2) Fusion, and 3) Reconstruction. During

the Decomposition stage, an image pyramid is formed for each of the images to be fused. The image pyramid, see Figure 2.1, is created by placing the original image at the base and then each successive layer is formed by filtering and down-sampling the previous layer's image. The successive filtering and down-sampling provide a multiresolution representation of the images. The next stage is Fusion. Fusion, in this sense, is the process of combining the information in the image pyramids to form a composite pyramid. Depending on the type of technique used, fusion can either be based upon the information in the image pyramids directly or it may require further decomposition of the image pyramids using various filters, see the right half of Figure 2.1. Reconstruction is the final stage. After the image pyramids have been fused into a composite pyramid, the process used to form the image pyramid is reversed to obtain the fused composite image. How well fusion is accomplished (in terms of Signal-to-Noise Ratio (SNRs) and image aesthetics) depends on: 1) the decomposition method, 2) the information used to determine how the image pyramids will be combined, and 3) the reconstruction method. How these affect fusion will be discussed below and experimentally demonstrated in Chapter III.

Three hierarchical image fusion techniques currently proposed in the literature [2,11,96] are described next. All three methods were implemented and tested during this research. A new image fusion algorithm, which was designed and implemented during this research, is presented in the next chapter. The new algorithm solves the problems associated with each of the three algorithms discussed below.

The first hierarchical image fusion technique presented is a method proposed by Toet [2, 93,94]. Toet's method is a multiresolution pyramidal technique that uses the maximum contrast information in what he calls the Ratio of Low Pass pyramids to determine what features are salient (important) in the images to be fused [2,93,94]. His method involves converting the input images into multiresolution ratio pyramids. The values in each pyramid represent what Toet calls the *contrast details*, where contrast is a measure of the difference in brightness across an image or scene. Next the ratio pyramids of contrast images are compared point-by-point and the maximum value is retained for the composite pyramid.

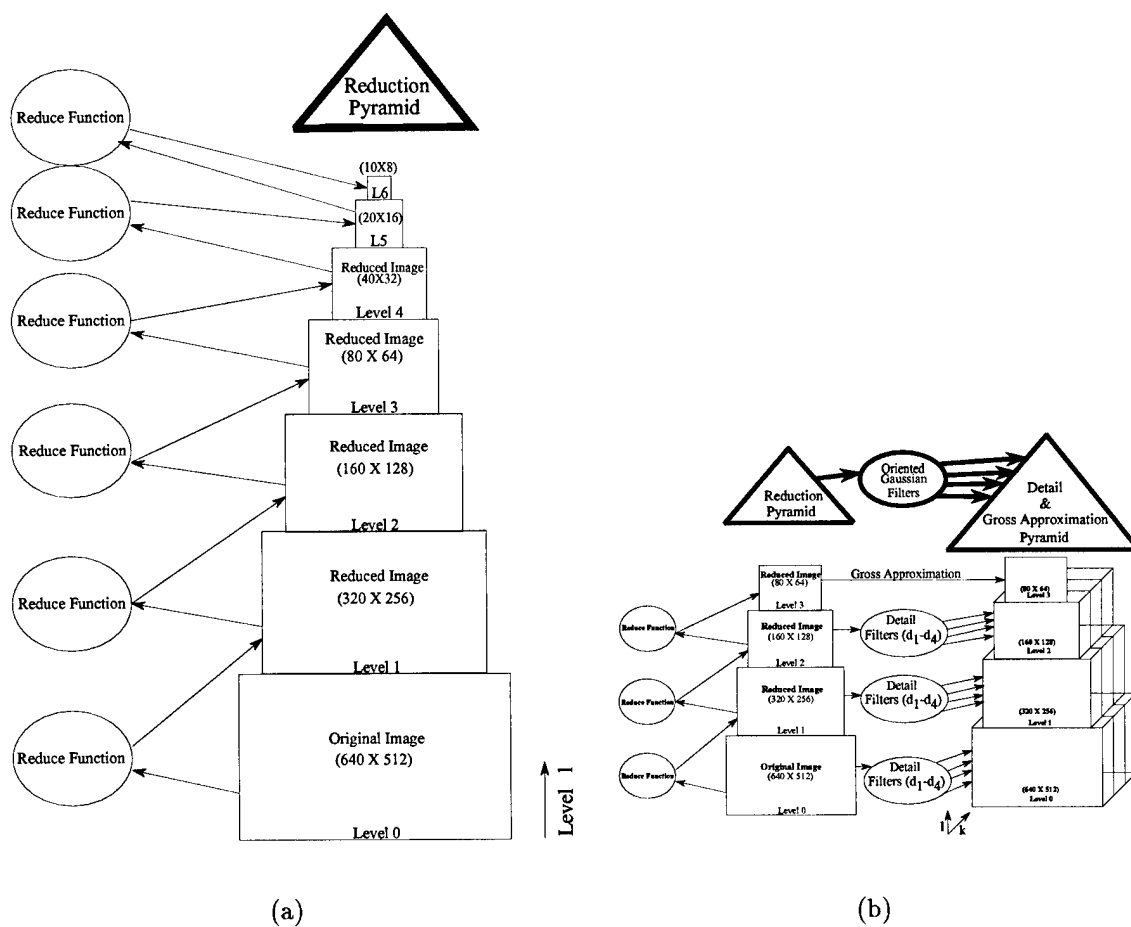


Figure 2.1 Figure (a) shows the first stage of a multiresolution pyramid algorithm, where each layer is a filtered and down-sampled version of the previous layer's image [96]. Figure (b) represents further processing of the image pyramid in (a).

Once the composite ratio pyramid is formed, the process of creating the ratio pyramid is reversed to recover the reconstructed composite.

Toet defends the decision to select the details for the composite image based upon maximum contrast by arguing that human vision is based upon contrast and that, by selecting details with maximum contrast, the resulting fused image will provide better details for the human analyst. Although this is sound reasoning based upon the desire to present the human analyst with the best visual image, it does not account for the fact that a noisy image is typically of higher contrast than an image that is not. Therefore, Toet's method would select the noisier parts of the images to be retained in the composite. Selecting the noisier parts for the composite presents a potential loss of information about desired targets. Thus, a method for selection that is based upon the perceptual sensitivity of the human visual system and not just pure contrast is needed.

The second method presented is a technique developed by Burt and Kolczynski [11,12]. Burt and Kolczynski's method is also a multiresolution pyramidal technique, but unlike Toet's method, Burt and Kolczynski have a more complex criteria to determine the salient features in the input images to be fused. First, they form the multiresolution image pyramid for each image to be fused. Next, they further decompose the image pyramid using four filters, called d_1, \dots, d_4 in Figure 2.1. The filters are used to extract orientation details from the multiresolution image pyramid. Once the details have been extracted, a match and saliency metric is used to determine which details are most important or salient. The match value compares two regions in the input images and provides a number that describes if they are similar, or match, and the saliency value represents the energy in the oriented gradients of the input images. If the input image details are similar (within some threshold) then the details are averaged [11,12]. If the input images are not similar enough (beyond the threshold) then the details from the image that are more salient are retained for the composite gradient pyramid. Once the fused gradient pyramid is formed, reconstruction is performed to recover the composite image.

The fusion algorithm proposed by Burt and Kolczynski has several advantages over the method proposed by Toet. Since it averages similar input sources, instead of just picking

some maximum value, it offers a potential for better noise reduction. It is important to note that the averaging is performed at the detail level, which eliminates the drawbacks associated with averaging at the pixel level. Burt and Kolczynski's method also allows the low contrast details to be preserved, if they are the salient features. The main disadvantage is that a template (weight matrix) is needed to decide which features are salient. Since there are always problems with size, orientation, translation, etc., finding a template that will work well as a salient measure, will be very difficult if not impossible. One possible weight matrix, proposed by Burt and Kolczynski, is a 3×3 matrix of ones; this allows the saliency to be based upon the local energy in the details. Although this provides a measure that has some useful applications, local energy in the details of some images may reflect a high value strictly due to noise in the image. Also, energy in the images that may not fall within human perception may play a large part in the decision of how the images will be fused. Therefore, a method that uses the strengths of Burt and Kolczynski's fusion scheme, and still addresses the human visual system, is needed.

The third method presented, which relies on the frequency response (contrast sensitivity) of the human visual system and uses the decomposition/reconstruction scheme developed by Burt and Kolczynski, was developed by Wilson [96]. Mathematically, contrast sensitivity (which is a measure of how a person responds to contrast at threshold) is defined as the reciprocal of contrast [61,84], where contrast, as defined above, is a measure of the difference in brightness across an image or scene. The measure of how the human visual system responds to contrast, i.e., contrast sensitivity, is a function of spatial frequency [61,84]. For example, it has been shown that the human visual system responds better, or is more contrast sensitive, to low spatial frequency components than it is to high spatial frequency components [84]. This is exhibited by the fact that contrast sensitivity peaks for frequency ranges between 2 and 10 cycles per degree (cpd) and then falls off sharply [61,84].

Wilson [96] uses contrast sensitivity to overcome the problem that Burt and Kolczynski had of determining a template to weight the details of the image pyramid. The idea is that by analyzing the frequencies in the multiresolution image pyramid and by weighting them according to the contrast sensitivity function of the human visual system, details that humans

would perceive as important would be preserved. While Wilson's method has been shown to provide better image fusion in terms of Signal-to-Noise ratios than either Toet's or Burt and Kolczynski's methods, it still suffers from a 2-3 percent reconstruction error based solely upon the decomposition and reconstruction method developed by Burt and Kolczynski. The fusion method designed and implemented for this research uses the experimentally derived contrast sensitivity function (CSF) of the human visual system developed by Mannos and Sakrison [50] and a wavelet-based decomposition/reconstruction method to overcome the limitations of the previously mentioned fusion techniques. A brief discussion on the wavelet decomposition and reconstruction method used during this research is presented next.

2.2 Wavelet Multiresolution Decomposition and Reconstruction

When discussing wavelets, it is helpful to relate them to other image analysis tools. One of the most well known is Fourier analysis. In Fourier analysis, a signal is represented as a weighted sum of sinusoids of different frequencies. For this research, the signal is a 2-dimensional image. The weights, which are called Fourier coefficients, provide information about the frequency content of an image. Thus, Fourier analysis is a mathematical method to transform the view of an image from a spatial representation to a frequency-based representation. The drawback to this method is that the Fourier transform provides information about frequency content present in a given image, but it does not tell where they occurred. To perform image fusion, it is important to compare images at the feature or local level [12]. Because the Fourier transform provides information at the global and not the local level it is not the method to use. One possible solution is to use a windowed Fourier analysis. In a windowed approach, the Fourier analysis is performed over a small region or window of the image. This provides both location and frequency content. The problem is choosing the window size. Once a window size is chosen, that window size is the same for all frequencies. The problem is that many signals (images) require a more flexible window size. For example, if the size is too small, information about low frequency content is lost. If it is too large, information about where the frequencies occur is lost.

Wavelet analysis provides the next step. Wavelet analysis is a windowed approach that allows the window size to vary as needed. It allows the window size to grow to obtain more precise information about the low frequency content of the image and it allows it to shrink for more precise position information about high frequency content.

This is accomplished by using a different form of basis function. In the Fourier or windowed Fourier approach, the basis functions are a set of periodic sinusoids. A wavelet-based approach uses a waveform of finite duration which has an average value of zero. In Fourier analysis the signal is represented as a sum of sine waves of varying frequencies. Wavelet analysis represents the signal as a sum of scaled and shifted versions of the original (mother) wavelet. Scaling has the effect of varying the window size. Shifting has the effect of moving the center of the wavelet across the image.

As with the Fourier analysis method, wavelet analysis uses an integral to calculate the coefficients of the transformation. For example, the one-dimensional Fourier transform is:

$$\mathcal{F}(\omega) = \int_{-\infty}^{\infty} f(t)e^{-j\omega t} dt. \quad (2.1)$$

which is the sum over all time of the signal $f(t)$ multiplied by the complex exponential, $e^{-j\omega t}$. The complete set of continuous Fourier coefficients is derived by allowing ω to vary over all frequencies. A similar approach is used for the continuous wavelet transform (CWT). The CWT is defined as the sum over all time of the signal $f(t)$ multiplied by scaled and shifted versions of the wavelet function Ψ :

$$C(scale, position) = \int_{-\infty}^{\infty} f(t)\Psi(scale, position, t)dt \quad (2.2)$$

In the CWT case, the complete Wavelet coefficients are derived by allowing the scale and the shift of the mother wavelet to vary continuously over all scales and shifts.

As with Fourier analysis, Wavelet analysis has both a continuous and discrete form. In the continuous case, the wavelet was allowed to vary over every possible scale and shift. In the Discrete Wavelet Transform (DWT), the scale and shift are varied based on powers

of two or a *dyadic*. Thus, each successive scale is twice the previous scale. The number of levels or scales chosen depends on the number of levels of analysis desired. In an image pyramid representation, the base level represents coefficients from the first or lowest scale and each higher level represents coefficients from a scale that is twice as coarse as the level below. The DWT discussed is the process of decomposing the image into its discrete wavelet transform coefficients. If the proper wavelet set is chosen, the image can be decomposed and resynthesized without reconstruction error.

The wavelet set or “mother” wavelet used in this research was the *Daubechies* 20-tap wavelet set [19]. It was chosen because it has perfect reconstruction (i.e., the decomposed image can be reconstructed from the wavelet coefficients without a loss of information). The Daubechies wavelet is generated from the scaling function $h(n)$ using Equation 2.3. The filter representing the scaling function is usually denoted as $h(n)$ and has a frequency response displaying low-pass characteristics. The filter representing the wavelet function is usually denoted as $g(n)$ and has a frequency response displaying high-pass characteristics. Equation 2.3 describes how $h(n)$ is used to generate $g(n)$. The scaling and wavelet functions corresponding to these filters are shown in Fig. 2.2. These waveforms are also used for reconstruction.

$$g(n) = -1^{1-n} \times h(1 - n); \quad (2.3)$$

The functions used for decomposition are actually the above mentioned functions reflected about zero and are shown in Figure 2.3. Notice that the scaling (\tilde{h}) and the wavelet (\tilde{g}) functions used for decomposition are not symmetric about zero. Figure 2.4 shows the frequency characteristics of the scaling and wavelet functions associated with \tilde{h} and \tilde{g} . Figure 2.5 shows the decomposition process for an image. The approximation and detail images are the result of different convolutional combinations of the scaling and wavelet functions. Figures 2.6 and 2.7 provide an example of a two level wavelet decomposition of a picture of Lenna [63]. Figure 2.6 represents the first level of decomposition and Figure 2.7 represents the second level. As can be seen in the figures, the decomposition process generates three sets of detail coefficients and one set of approximation coefficients. The only information

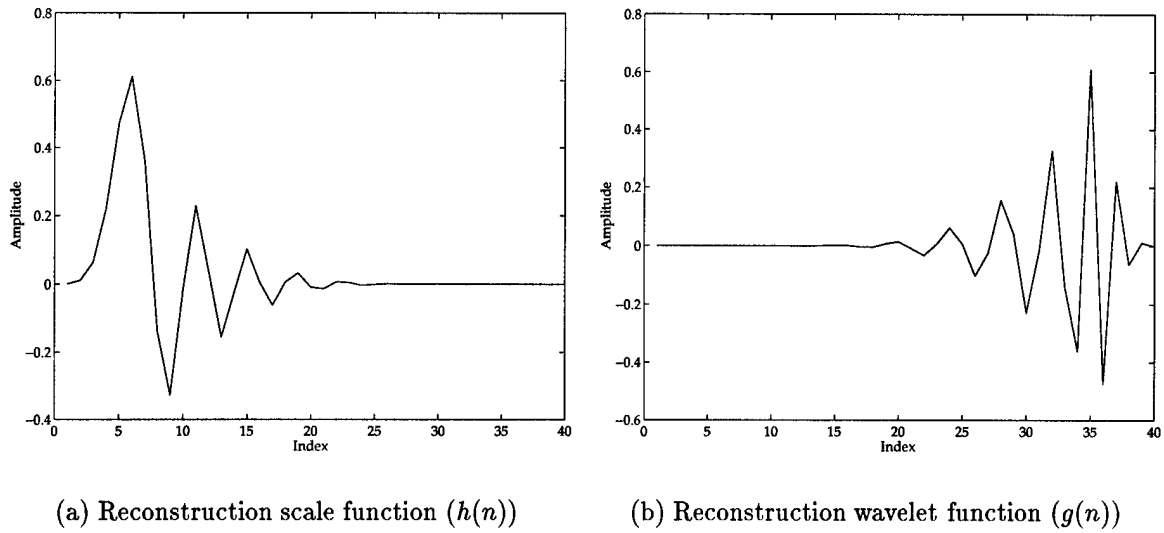


Figure 2.2 Reconstruction Scale and Wavelet Functions.

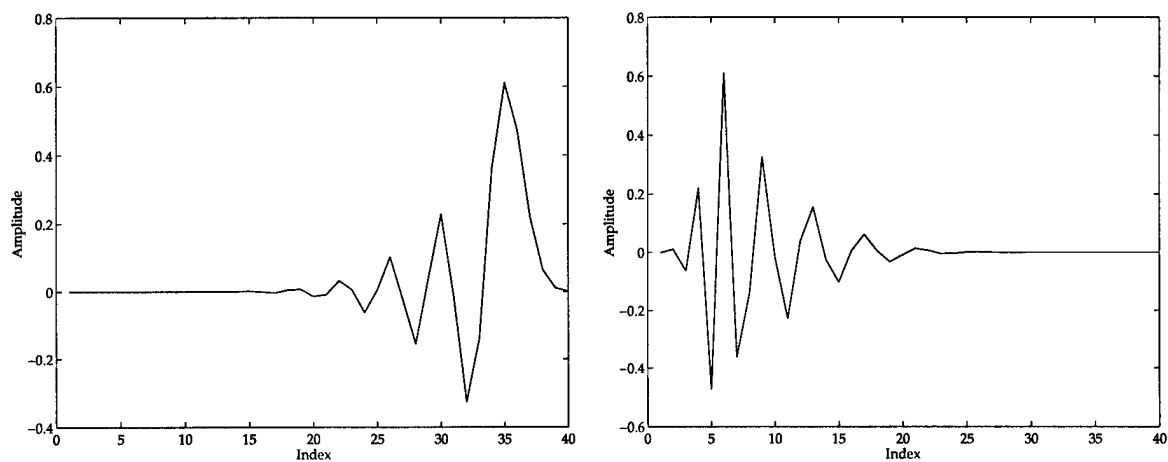
needed for reconstruction is one set of approximation coefficients from the highest decomposition level and all of the detail coefficients from every level. The approximations from the lower levels of the image pyramids are not needed.

Reconstruction of the original image is the mirror operation of decomposition. Figure 2.8 shows the process of reconstructing an image after wavelet decomposition.

How the Daubechies DWT applies to hyperspectral image fusion will be discussed in the next chapter, when the novel physiologically motivated fusion algorithm developed for this research is described. Now that several hierarchical image fusion algorithms have been discussed and the limitations of each have been detailed, the next step is to discuss possible algorithms that can be used to classify the outputs of the fusion process.

2.3 Artificial Neural Network Architectures

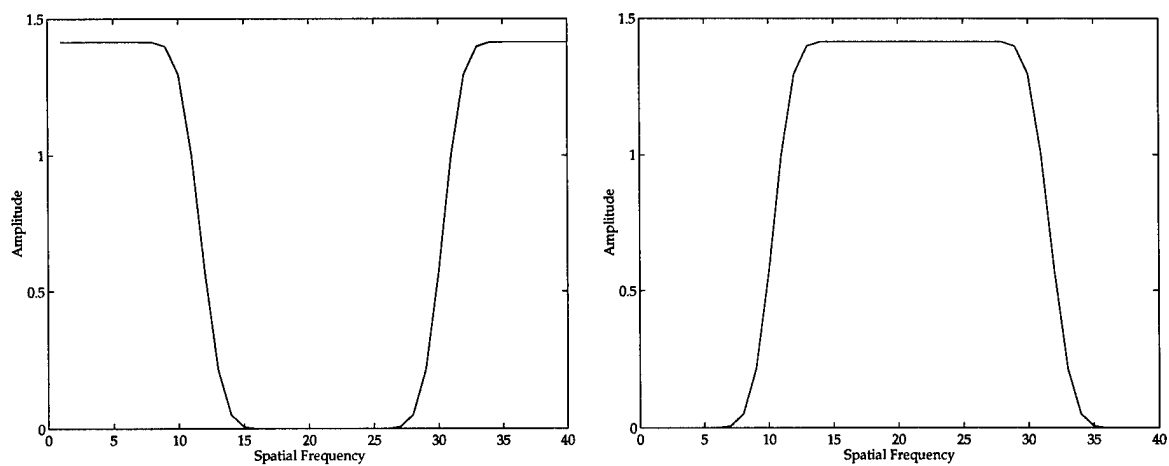
As mentioned in the introduction, Artificial Neural Networks (ANNs) have been successfully applied to many types of classification problems, from breast cancer detection in digitized mammograms to target detection in infrared imagery [43,47,65,73,74,82]. Again, the reason they are so successful is that they can be trained to correctly classify previously unseen test samples based upon the information in a set of training exemplars. How well



(a) Decomposition scaling function ($\tilde{h}(n)$)

(b) Decomposition wavelet function ($\tilde{g}(n)$)

Figure 2.3 Decomposition Scale and Wavelet Functions.



(a) Wavelet high-pass filter

(b) Scale low-pass filter

Figure 2.4 Wavelet frequency responses.

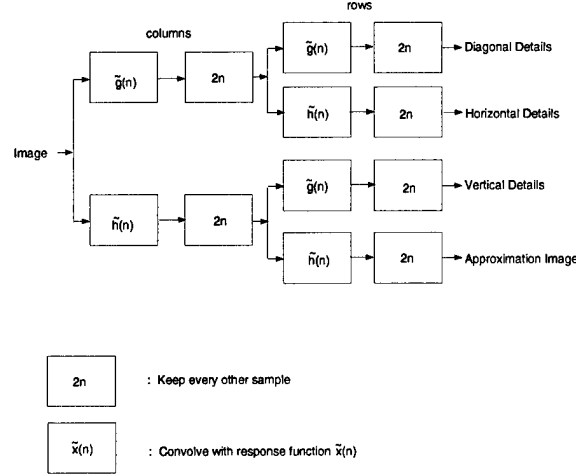


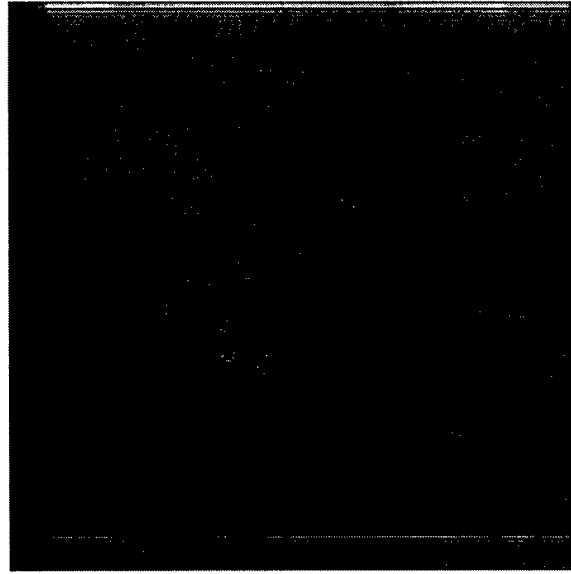
Figure 2.5 Decomposition flow chart to obtain first level of decomposition [49].

they perform classification on the test data is a function of the training data, the training algorithm, and the network architecture. While it is shown in the literature that the amount of training data and whether or not the distribution of the training data represents the distribution of the test data are critical to the test performance of a trained ANN [4–6, 25, 95], the focus of this discussion will be on the ANN architecture.

2.3.1 Fixed Architecture Feed-Forward ANNs. In a feed-forward ANN, the flow of the network is from the inputs to the outputs without any feedback loops [85], as shown in Figure 2.9. Thus, the outputs of a trained network are strictly a function of the present inputs and not any previous outputs from any layer. Figure 2.9 depicts a feed-forward network with one hidden layer. Cybenko [17] showed that a Multilayer Perceptron (MLP) using only a single hidden layer of sigmoidal hidden units could approximate any Lebesgue integrable function to an arbitrary closeness. Park and Sandberg [59] and Hartman [30] showed that an RBF network using a single hidden layer of Gaussian hidden units could approximate any Lebesgue p -integrable (L^p , for $1 \leq p < \infty$) function to an arbitrary closeness. Therefore, the class of functions that can be approximated by RBF networks are a superset of those that can be approximated by MLPs. This is why RBF networks, using Gaussian hidden layer nodes, are called “Universal Function Approximators” [9, 30, 59] and why an RBF architecture was chosen for this research.



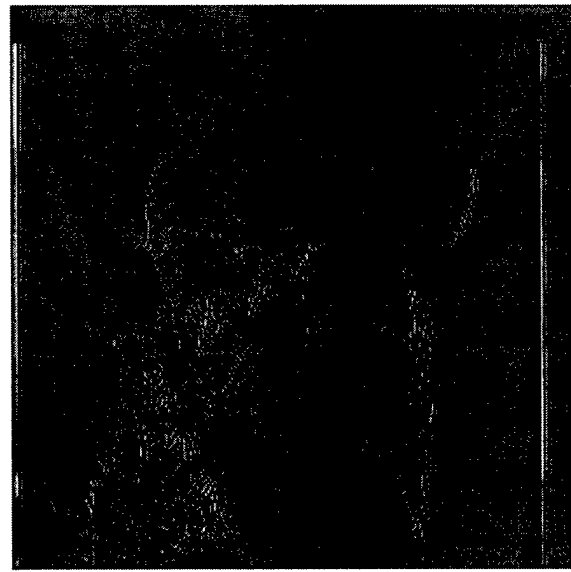
(a) Approximation coefficients



(b) Horizontal Detail Coefficients

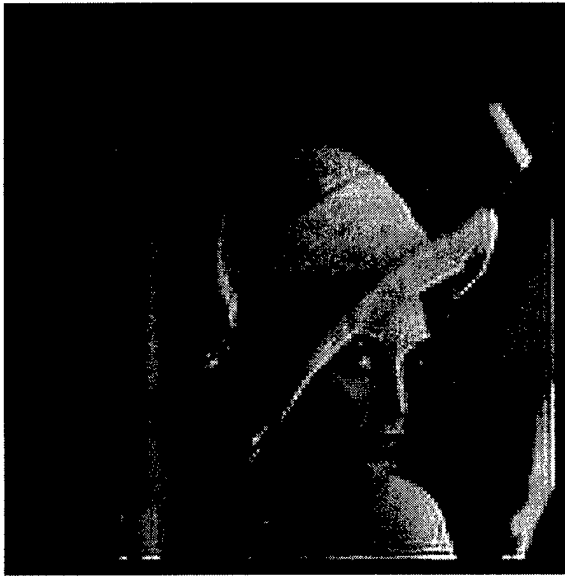


(c) Diagonal Detail Coefficients

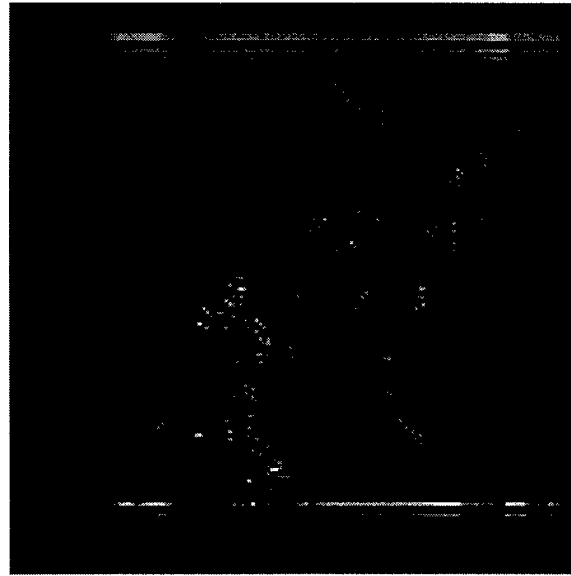


(d) Vertical Detail Coefficients

Figure 2.6 This figure represents an example of the first level of a DWT decomposition of a picture of Lenna.



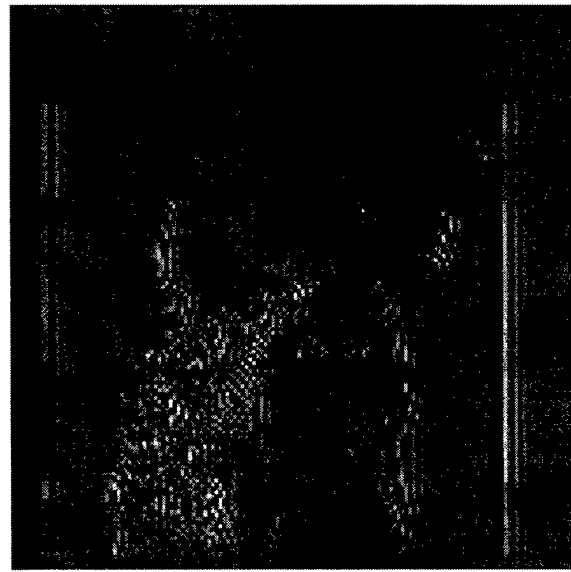
(a) Approximation coefficients



(b) Horizontal Detail Coefficients



(c) Diagonal Detail Coefficients



(d) Vertical Detail Coefficients

Figure 2.7 This figure represents an example of the second level of a DWT decomposition of a picture of Lenna.

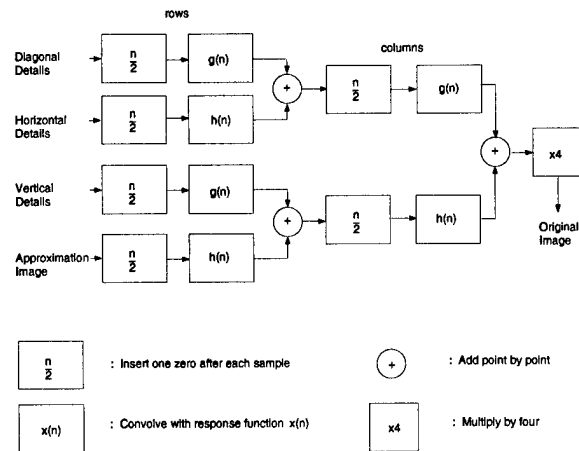


Figure 2.8 Reconstruction of an image after a single level of decomposition [49].

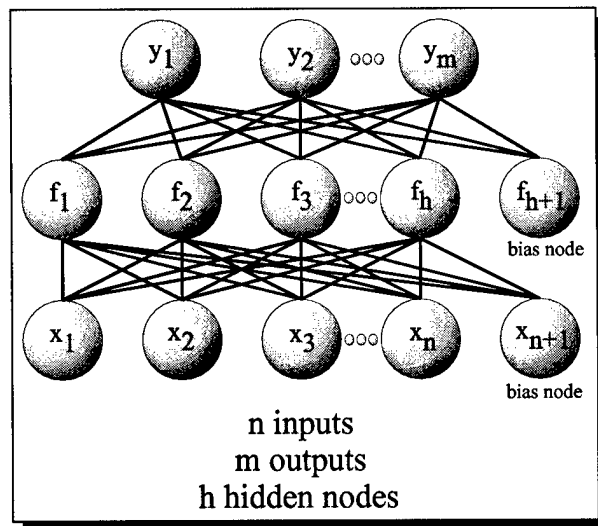


Figure 2.9 Typical Feed-forward ANN Architecture [83].
This Figure depicts the typical architecture for feed-forward ANNs [83].

RICA is an RBF neural network that uses Gaussian hidden layer nodes. Therefore, it has the property of Universal Function Approximation. As a result of this inheritance, RICA has the ability to approximate virtually any family of functions, including anything representable by an MLP. Stating that a class of ANN architectures have the ability to approximate a family of functions to an arbitrary closeness only provides part of the story. Determining the size of the network and the parameters providing the closest approximation is the rest of the story.

In most of the earlier work in feed-forward neural networks, the focus was on algorithms that optimized the parameters of the ANN and not on algorithms that optimized the network size. The size of the network (number of hidden layer nodes) was typically selected through trial-and-error or experience with a particular data set [23,24,45]. In either case, the number of nodes was preselected and fixed by a human. Once the size of the architecture was selected, the network parameters were optimized to find the best set of parameters for the given architecture, where best is defined in terms of function approximation or classification accuracy. This type of network optimization is associated with fixed architecture learning (i.e., the size of the architecture is fixed before training begins).

In fixed architecture learning, the goal is to find a set of parameters that minimize the difference between the function represented by the ANN and the desired function to be learned. The desired function may be a discriminant function that allows the ANN to perform classification on a given set of data. For MLPs, the parameters are the weights on both the hidden and output layer nodes, the biases used in the activations of the sigmoids [28,75], and a parameter that changes the slope of the sigmoid (this parameter is typically fixed at 1). For RBFs, the parameters are the centers and variances of the Gaussian hidden nodes and the weights on the output layer nodes [9,53]. Methods for finding MLP and RBF parameters are discussed next.

2.3.1.1 Optimization Using Back-propagation and Gradient Descent. One of the most popular methods for training MLPs and one that has been shown to be equally effective in training RBFs is back-propagation [9,75,83,85,88]. Back-propagation derives its name from its method of parameter optimization. In an over-simplified view, back-

propagation seeks to minimize the error between the desired output of the ANN and the actual output of the ANN by back-propagating the error from the output nodes to the input nodes and modifying the network parameters to minimize the error produced by the previous layer's outputs.

Minimization through back-propagation is performed by taking the derivatives at each layer with respect to the weights and then setting those derivatives equal to zero to get the necessary conditions for the optimal weights [9,75,85,88]. The network parameters (weights) are then modified to solve the system of equations.

There are iterative minimization techniques called Gradient Descent algorithms [9,23,31,66,75,85,88] that are used to minimize the back-propagated errors. Gradient descent algorithms work on the principle that minimization can take place if the parameters are modified in the direction of the negative gradient. The negative gradient is the direction that will minimize the above-mentioned derivatives with respect to the weights of the neural network for MLPs and the centers, variances, and weights for RBFs. The weight updates or modifications can occur after each training sample is presented to the network. This is termed an *instantaneous training* method. Once all of the training samples have been presented to the network once and the updates have occurred, it is said that an epoch of training has been accomplished. The *batch training* method only modifies the weights after the entire set of training samples have been presented to the network and the cumulative error has been calculated. While back-propagation using gradient descent has been shown to be very effective in training ANNs, it has some serious drawbacks. First, many thousands of epochs may be needed to train the ANN to a desired accuracy level. If the training set is large, then the number of Floating Point Operations used to calculate each weight/parameter can be overwhelming in terms of computing time. The second drawback to back-propagation with gradient descent is that a global minimum is not guaranteed to be found. That is, the set of network parameters that may achieve a minimal error (mean squared-error, sum squared-error) is not guaranteed to be found.

2.3.1.2 Optimization Using Supervised/Unsupervised Clustering. RBF networks attempt to overcome the problems associated with back-propagation and gradient

descent by training the hidden layer nodes separately and then training the output layer weights [53]. Many RBF networks train the hidden layer nodes using some variant of the unsupervised Kohonen Self Organizing Feature Map (SOFM) [35,38,41] to cluster the input training data [9,16,92]. The clustering is then used to find the centers of the Gaussian hidden layer nodes (each center is represented by the weight vector associated with each of the nodes in the Kohonen SOFM).

It has been widely reported that the distribution of the SOFM nodes represent the probability density of the input data [26,35,38,41] and that the centers of the Gaussian units, which are derived from the SOFM nodes, will also represent the probability density of the input data [9,16,92]. The accurate modeling of the probability density by the hidden layer nodes would then provide for optimal training of the output layer weights separately to approximate the *a posteriori* outputs of the RBF. While this is the desired outcome, it has recently been shown that the distribution of the nodes in a Kohonen SOFM do not necessarily represent the true probability density of the data [100]. For example, in the ideal case, the nodes of a Kohonen SOFM would be distributed such that each node has an equal probability of winning. A node "wins" if the Euclidean distance between its center and a random sample x is less than any other node. Thus, in the "ideal" more nodes are placed where the data are dense than where the data are sparse. Experiments conducted in this research indicate that the frequency of winning for Kohonen SOFMs is not uniform across the nodes and therefore it does not represent the true probability density of the data, see Figures 2.10, 2.11, and 2.12. The left side of these figures show the results of forming the Kohonen SOFM and the right side uses circles to indicate the frequency of winning. Each circle represents the frequency of winning for a given node. The larger the radius the more frequently the node wins. It is easy to see that some nodes win much more frequently than others and that there are nodes in regions where no training data were present. Therefore, RBF networks which use this popular form of network training may not be truly optimizing the network.

Other methods, both unsupervised and supervised (HEC [51], K-Means [20], LBG [46], Fuzzy C-Means [57], and LVQ [40]) may also be used to define the centers of the Gaussian

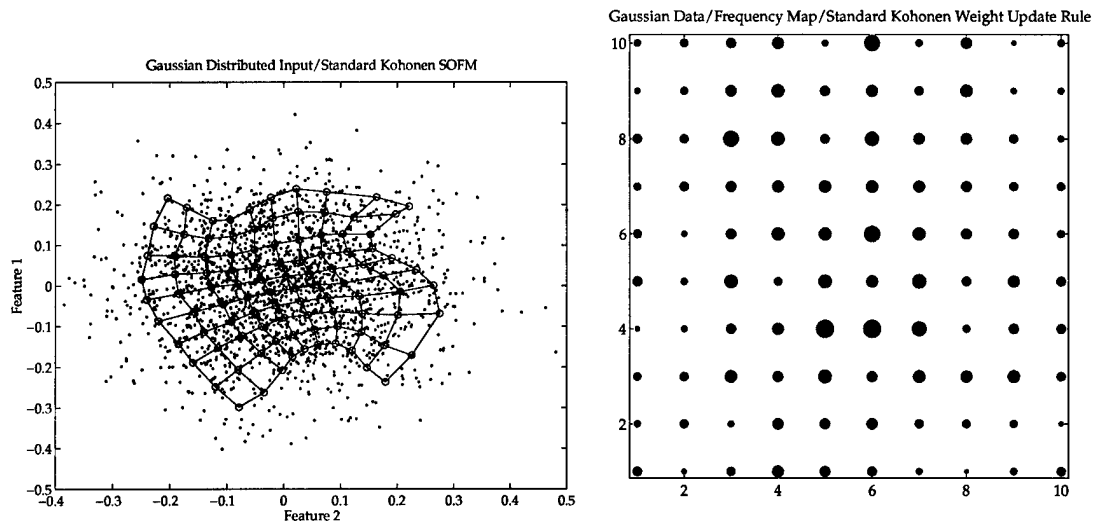


Figure 2.10 The figure on the left represents clustering using the Kohonen SOFM. The figure on the right uses the radius of a circle to represent the frequency of winning for each node.

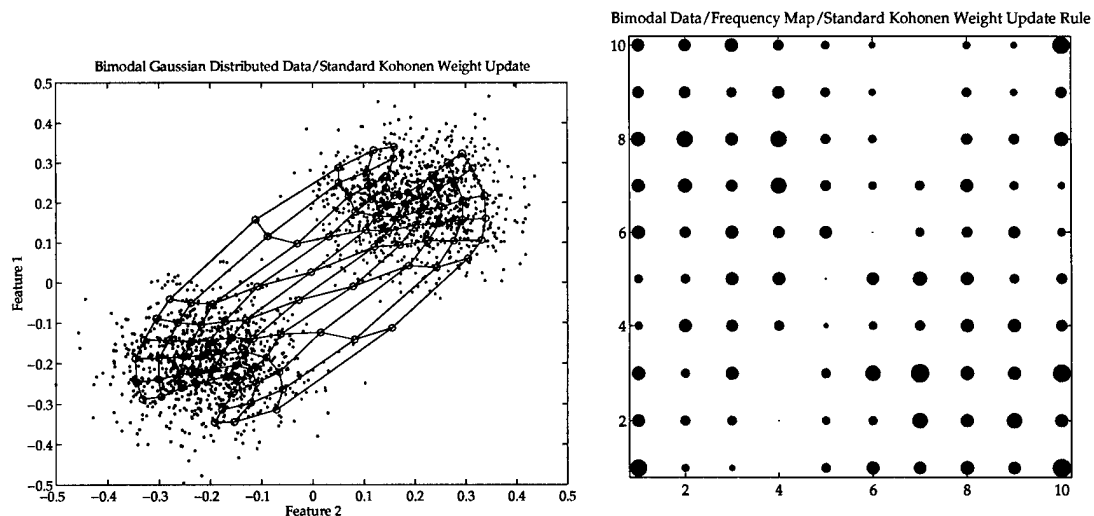


Figure 2.11 The figure on the left represents clustering using the Kohonen SOFM. The figure on the right uses the radius of a circle to represent the frequency of winning for each node.

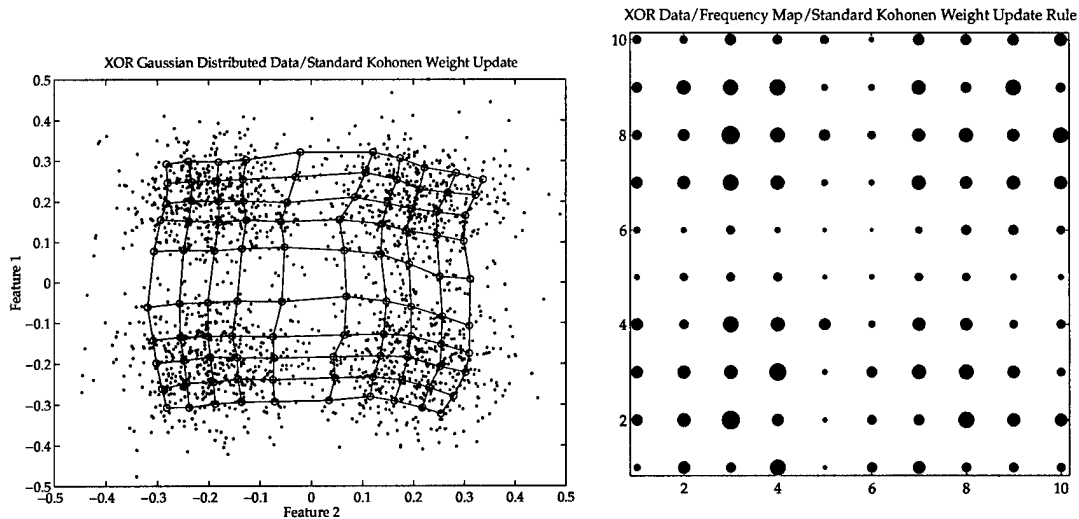


Figure 2.12 The figure on the left represents clustering using the Kohonen SOFM. The figure on the right uses the radius of a circle to represent the frequency of winning for each node.

hidden units. These methods suffer from numerous problems, such as determining the number of clusters needed, their shapes, their volume, etc [98,99].

The clustering techniques described above are used to find/define the Gaussian hidden node centers. The variances of the hidden nodes are calculated using various other methods. Some methods use a common variance for all nodes and rely on the training of the output layer weights to overcome any limitations imposed by such an assumption [14]. This works if the clusters at the hidden layer are similar in shape. If not, there will be errors associated with the mismatch in the approximation of the data by the improperly scaled Gaussian. If the variances are chosen to be too small, there will be regions that are not covered by the Gaussian hidden units. If the variances are chosen to be too large then the Gaussian hidden units will overlap too much and the loss of fidelity will result in network errors.

Some people attempt to overcome the one-size-fits-all variance problem by allowing the variance to change from node to node. They choose a variance that is based upon the Euclidean distance between a node and its neighbors [27,36,53]. The variance is then adjusted until the outputs of the Gaussian hidden nodes are equal at the halfway points between the node centers. This works if the distribution of the data points around the adjacent nodes are

similar in variance, but if they are very different, one or both of the Gaussians may cover a region of the feature space that contain no data points, or a disproportionately large number of them. This problem is similar to one associated with using Kohonen SOFMs. Sometimes nodes are moved into a region that contain no data points and therefore do not represent the probability density of the data. RICA was designed to overcome the problems stated above by allowing the centers and variances of the Gaussian hidden nodes to vary according to the distribution of the data.

After the hidden layer node parameters have been calculated, the next step in training RBF networks is to train the output layer weights. There are several minimization techniques used to train the output layer weights which include some form of statistical risk or cost minimization [9, 42], but the most popular techniques use a pseudoinverse to perform a squared-error minimization [9]. One of the disadvantages of using the pseudoinverse to perform a squared-error minimization is that while it produces a set of weights that produce a minimum squared error at the output, it is not necessarily the best solution for classification. The next section discusses the standard pseudoinverse method and the Ho-Kashyap algorithm [20, pp 159-166]. Ho-Kashyap is the iterative minimum squared-error method that RICA uses to overcome the limitations of the pseudoinverse method.

2.3.1.3 Optimization Using MSE and Ho-Kashyap Methods. In many RBF algorithms the output layer weights for each output node are determined using the following minimum squared error (MSE) procedure [9]. Let $y_{m,n}$ be the desired output of node m for input sample x_n , h_n be the output of the hidden layer nodes for input x_n , and w_m be the weight vector associated with output node m . If $n = 1, 2, \dots, N$, where N is the number of training samples, then

$$y_{m,n} = w_m \cdot h_n$$

defines a system of linear equations that can be used to find the weight vector w_m for each output node $1, 2, \dots, M$. The MSE procedure involves finding a minimum squared error estimate to the above system of linear equations. Rewriting the above system of linear equations in matrix form yields:

$$Ya = b,$$

where Y is the $N \times (H + 1)$ augmented matrix that is formed by creating a matrix with rows corresponding to the hidden layer output vector h_n associated with each input vector x_n for all samples, and then augmenting with a column of ones. H is the number of HLN's, b is the $N \times 1$ column vector whose component values are the desired outputs for each input, respectively, and a is the weight vector we wish to calculate. Augmentation permits a solution vector w_m that does not have to pass through the origin. As mentioned above, the desired output is 1 for the sample of the class corresponding to the output node under training and -1 for all other classes.

If Y were invertible then the direct solution would be $a = Y^{-1}b$. Typically there are more samples than there are hidden layer nodes (i.e., $N \gg H$), therefore the Y matrix is not square and is not invertible. Using the pseudoinverse the solution is $a = Y^\dagger b$, where Y^\dagger is the pseudoinverse and is defined as $(Y^t Y)^{-1} Y^t$, where $(Y^t Y)$ is invertible and t denotes the transpose operator. While this method is guaranteed to find a minimum mean squared-error (MSE) solution, if the data are linearly separable, it is not guaranteed to find the separating vector w [20]. The reason it may not find the separating vector is that the wrong margin vector b may have been chosen. The Ho-Kashyap method, which is the method RICA uses to train the output layer weights, is designed to overcome the shortfalls of a simple pseudoinverse for finding the solution vector a .

Back-propagation is a method by which all of the network parameters can be optimized simultaneously. It was also shown that RBF networks and MLP networks could optimize the hidden and output layer parameters separately. The Ho-Kashyap method is an iterative algorithm used to optimize the output layer weights of an ANN. The reason the Ho-Kashyap method is used to calculate the weights for RICA is that it is specifically designed to find the separating vector between linearly separable data in a finite number of iterations. It is also guaranteed to converge to a solution even if the input data are not linearly separable. If the data are linearly separable, the algorithm will terminate with an error vector e_k , which is presented in detail in Section 4.2.1, equal to zero. If the data are not linearly separable, it will still converge to the MSE solution as with the pseudoinverse method, but the error

vector e_k will be less than zero. This is important because the proof presented in Section 4.3 uses the fact that RICA is a MSE algorithm to show that it performs a minimum mean squared-error approximation to a Bayes optimal classifier.

The reason the Ho-Kashyap method is guaranteed to find the separating vector is because of the unique way that it modifies both the weight vector a and the solution vector b during its training [20, pp 159-166]. To perform the Ho-Kashyap method the Y matrix defined above is modified slightly. The Ho-Kashyap method solves the system of equations defined by $Ya = b$ as above, but now the components of b are constrained to be greater than zero and b also changes during training. In order to make all of the components of b greater than zero the same Y matrix defined above is used with the following modifications. The rows corresponding to the input vectors which would have had a -1 in the b vector above, are multiplied by -1 . Now b can start with the same value for all components. In effect, we did not change the system of equations, we simply multiplied through on both sides by -1 for the input samples that were from a class other than the one corresponding to the particular output node. For example, given the following Y matrix:

$$Y = \begin{bmatrix} .5 & .2 & .3 & .7 & 1 \\ .5 & .8 & -.1 & .3 & 1 \\ .1 & .3 & .2 & -.2 & 1 \\ -.1 & .8 & .7 & .9 & 1 \\ .3 & .6 & .9 & .1 & 1 \\ -.1 & -.2 & .4 & .5 & 1 \end{bmatrix} \quad (2.4)$$

where the first four columns represent the training data, the fifth column is the ones vector used to augment the data and the first through third rows are inputs from the class corresponding to the output node under training. After modifying Y to use it in the Ho-Kashyap method it becomes:

$$Y^* = \begin{bmatrix} .5 & .2 & .3 & .7 & 1 \\ .5 & .8 & -.1 & .3 & 1 \\ .1 & .3 & .2 & -.2 & 1 \\ .1 & -.8 & -.7 & -.9 & -1 \\ -.3 & -.6 & -.9 & -.1 & -1 \\ .1 & .2 & -.4 & -.5 & -1 \end{bmatrix} \quad (2.5)$$

The fourth through sixth rows (which represent a class other than the one for the node being trained) have been multiplied by -1 . As discussed above, for the MSE procedure, if we pick an arbitrary margin vector b , then all we can say about the MSE procedure is that it minimizes $\|Ya - b\|^2$. If the input samples are linearly separable then there exists some values \hat{a} and \hat{b} such that

$$Y\hat{a} = \hat{b} > 0,$$

where $\hat{b} > 0$ means that every component of \hat{b} is greater than zero. The solution \hat{a} is the separating vector. The problem is selecting the proper margin vector \hat{b} . We do not necessarily know \hat{b} *a priori*. The Ho-Kashyap method modifies the MSE procedure to find both the \hat{a} and \hat{b} .

The basic idea is that if we minimize the following criterion function subject to the constraint that $b > 0$ and we allow both b and a to vary, then the minimum of the criterion function will be zero and the solution a will be the separating vector.

Defining the criterion function $J(a, b)$ as:

$$J(a, b) = \|Ya - b\|^2,$$

the gradient of $J(a, b)$ with respect to a is

$$\nabla_a J(a, b) = 2Y^t(Ya - b)$$

and the gradient of $J(a, b)$ with respect to b is

$$\nabla_b J(a, b) = -2(Ya - b).$$

For any given vector b we can choose a vector a such that $a = Y^\dagger b$. This yields $\nabla_a J(a, b) = 0$, which minimizes $J(a, b)$. While a depends on b and is free to take on any set of values, we must constrain the vector b to be greater than zero. This is accomplished in the Ho-Kashyap method by starting with $b > 0$ and by not allowing the value of b to be reduced.

Since the Ho-Kashyap method uses a gradient descent procedure to minimize the criterion function $J(a, b)$ with respect to both a and b it needs to consider the constraints on b when performing its updates. To perform a search along the negative gradient for a solution vector \hat{b} and to maintain the constraints on b , the Ho-Kashyap method must only allow positive changes to the components of b that are in the negative gradient direction. For example, typical gradient descent algorithms would define the update to b as follows:

$$b_{i+1} = b_i - \rho [\nabla_b J(a_i, b_i)],$$

where i is the iteration number and ρ is a scaling factor (typically $0 < \rho \ll 1$). This update method allows the components of b to be reduced as well as increased. We want to ensure that $b > 0$ therefore, we can only allow b to be increased. The way we will ensure this is to not allow any negative changes in components of b . This yields a gradient descent update to the vector b defined by:

$$b_{i+1} = b_i - \rho \frac{1}{2} [\nabla_b J(a_i, b_i) - |\nabla_b J(a_i, b_i)|].$$

This update zeros out any changes that would be negative. Using the above discussion and equations, we define the Ho-Kashyap algorithm in Section 4.2.1.

In all of the methods discussed so far, the goal has been to optimize a set of parameters for a given network architecture. While the individual drawbacks of each of the training methods has been discussed, the one common weakness shared by all is that their architectures had to be chosen *a priori* by a human. To be truly useful and to mimic a biological neural network, the ANN must be able to autonomously adapt its architecture to provide the best possible test accuracy for a given data set [78]. Network algorithms that modify both the network parameters and the network architecture are called “*Adaptive ANNs*”. RICA belongs to this class of ANNs.

2.3.2 Adaptive Architecture Feed-Forward ANNs. When designing algorithms that autonomously adapt the network architecture it is important to keep in mind that the size of the architecture is critical to the ability of the network to perform function approximation. If the architecture has too few hidden nodes, the network may not have enough function representational power to reasonably approximate the necessary discriminant function. In other words, there may not be a combination of the chosen number of hidden nodes that can adequately approximate the underlying distribution of the data. If there are too many hidden nodes, there may be too many free parameters in the network and not enough data to adequately define them [5, 6, 25, 95]. Too many free parameters will most likely prevent the neural network from generalizing [7, 9, 25, 27, 95] because it memorizes the training data and is so over-specified it does not perform well on the test data.

Networks that are designed to adapt their architectures to fit a given data set commonly use two approaches: pruning and construction. Pruning techniques start with a fixed architecture size and then remove nodes or weights until a desired fitness or output is achieved [3, 44, 69, 70, 91]. Construction algorithms start with one or a small number of nodes and selectively add nodes until some desired criteria is met: ART, ARTMAP [13, 45, 54], MLP [15, 22–24, 52, 56, 68, 90, 91], and RBF [8, 27, 36, 37, 55, 78].

2.3.2.1 Pruning Techniques. Le Cun [44] presented a method of MLP pruning entitled “Optimal Brain Damage” in which the weights of a neural network were analyzed at given points in the training process. Any weights that were found to be unimportant, based upon some saliency metric, were forced to zero and were not updated in subsequent training. Steppe [91] also introduced a pruning method to determine an MLP’s architecture. Her method involved a two stage statistically rigorous approach to architecture selection. First several MLPs are trained using a fixed number of preselected hidden layer nodes. The networks that achieve a desirable error rate, based on a statistical analysis of the entire set of trained networks, are then used in the second stage. The second stage involves removing a node and then retraining the networks. After node reduction and retraining, a statistical analysis is performed to determine if the performance of the reduced networks is statistically equivalent or better than the non-reduced networks. If it is equal or better, another node

is removed and the retraining and analysis is repeated. If it is not, pruning stops and the network architecture has been selected. Two main differences between Le Cun's, Steppe's, and other's pruning techniques is the method used to determine the number of initial hidden layer nodes and the method used to determine which weights or nodes should be eliminated.

There are several problems associated with designing an ANN architecture based upon pruning techniques [69]. The first is in determining the number of nodes with which to start. If too few are chosen, the network will not have enough nodes and will suffer from the problems discussed above. If too many are chosen and the network over-learns, subsequent training, even after pruning, may not compensate for the over-learning. A second problem is that pruning techniques do not allow for weights that may be correlated. The effect one weight has on the network may be highly correlated with another weight in the network [69]. A third problem is determining the saliency metric to use. Ruck [80] and Priddy [67] have provided ways to analyze the saliency of the weights, but both require a trained ANN. The training, saliency analysis, weight removal, and subsequent retraining is computationally expensive. A final problem is determining the rate at which to analyze and remove the weights. Choose a rate that is too fast and the network may not adequately explore the weight space [70]. Choose a rate that is too slow and convergence to a solution may take a very long time [70]. Steppe's method [91] overcomes many of the problems associated with pruning, but her method is computationally expensive. The bottom line is that there are a considerable number of problems to overcome when using pruning techniques. This is why the method used to train RICA is a constructive algorithm and not a pruning algorithm.

2.3.2.2 Constructive Techniques. In general, constructive algorithms infer their architectures from some form of partitioning of the training data. The type of partitioning is often based upon the type of hidden layer function used. ANNs that use sigmoidal hidden layer functions will often partition the training data based upon the separating boundary between adjoining class pairs [15,23,52,68]. This is because sigmoidal functions act as separating hyperplanes. ANNs that use Gaussian or Ellipsoidal hidden layer functions, will often use an unsupervised/supervised clustering technique to partition the input training data [8,27,36,37,55,78]. Again, this is because clustering algorithms partition the data into

groups that can be represented by the Gaussian or Ellipsoidal functions. Adaptive Resonance Theory networks (ART and ARTMAP) [13,54] use templates to partition the input feature space into hyperrectanguloids.

The idea is that by partitioning the data along the separating boundaries, constructive algorithms can modify their architectures as well as their parameters to estimate the underlying probability density of the data. Fixed architecture algorithms also sought to estimate the underlying probability density of the data, except only the network parameters were modified and not the network architecture. Thus, for constructive algorithms, the problem becomes one of finding the separating boundaries, adding enough hidden layer nodes, and parameterizing the hidden layer nodes to truly represent the probability density of the training data.

There are many techniques used to determine if enough nodes have been added. Some involve starting with a small number of nodes and training the network until some criterion function is minimized. Then a node is added and the network is retrained. The process is repeated until some (often arbitrary) stopping criterion is met. A useful stopping criteria is to keep a validation set separate from the training set and to train and test until the test accuracy on the validation set decreases. This is a valid method to optimize an ANN, however it has some problems. One, the retraining takes a lot of CPU time. Two, research has shown that in some cases, further node reduction or increase may have resulted in further improvement in the test accuracy, even after a drop was experienced [90,91]. Steppe's underlying theme was that analysis of a neural network architecture was a stochastic process and many networks using the same size architecture needed to be trained and analyzed before the capability of a particular realization could be quantified.

Another method used by constructive algorithms is to continue to add nodes until all of the training samples have been separated [22-24,68]. This may guarantee 100 percent accuracy on training data, but it does not guarantee that the network will generalize well. The network may be over-specified. Additionally, adding more nodes does not guarantee that a network will perform better. Adding more nodes in a network may cause the network to suffer from the same "Curse of Dimensionality" that adding more and more features to

a data vector can cause [7, 29]. As these nodes are added, the network needs exponentially more data to adequately represent the feature space.

While adaptive algorithms have been designed to overcome the need to preselect the number of nodes, their methods of partitioning the input feature space are largely *ad hoc* and they still suffer from the same problems of optimizing the network parameters as fixed architecture algorithms. RICA is an adaptive algorithm that overcomes the partitioning problem by using a Gaussian goodness-of-fit technique to determine the number of hidden layer nodes. It uses a Mahalanobis distance clustering algorithm to overcome the problems of cluster size, shape, and location. Additionally, RICA's uses the Ho-Kashyap algorithm [20, pp 159-166], which is discussed in Section 2.3.1.3, to train the output layer weights. The Ho-Kashyap algorithm overcomes the problems associated with using a standard minimum squared-error method to determine the output layer weights.

2.4 Summary

This chapter described current hierarchical image fusion and ANN techniques and pointed out why they are not appropriate to address the problems defined for this research. The next chapter describes the novel hierarchical image fusion algorithm which was designed and implemented to fuse hyperspectral image data and to overcome the problems associated with current techniques.

III. Wavelet-Based Hierarchical Image Fusion with Contrast Sensitivity Saliency

The purpose of this chapter is to describe the physiologically motivated image fusion algorithm that was designed and implemented during this research. Condensed versions of the contributions of this discussion have been published in Optical Engineering [97] and in IEEE's Transactions on Geoscience and Remote Sensing [96]. The chapter is organized to provide background on the algorithm's design, details of its implementation, and experimental results. The results support the conclusion that this new method is superior to previous fusion methods in both signal-to-noise ratios and visual aesthetics.

3.1 Background

Before the details of the algorithm are presented, it is important to discuss the reasons for using a combination of contrast sensitivity and wavelet-based decomposition/reconstruction. The question raised is actually two-fold:

1. Why use contrast sensitivity?
2. Why use wavelets?

Why use contrast sensitivity? One reason is that because humans are very good at detecting targets in image data and the goal of this research is to automate target detection, a logical approach is to model the human system. In addition, it has been shown by Stager and Hameluck that contrast sensitivity was better in determining a person's ability to perform air-to-ground search than standard visual acuity tests [89]. Thus, contrast sensitivity is an important part of how people perform target detection. Another reason to use contrast sensitivity is that the CSF is mathematically tractable, has a direct relationship to the frequencies in an image, and can easily be implemented in computer software. Finally, contrast sensitivity has already been shown to be effective in image fusion and noise suppression [96].

The next question is why use wavelets? As discussed in the previous chapter, one of the limitations with current image fusion techniques is the type of image decomposition and

reconstruction performed. In all of the previously mentioned fusion methods there was at least a 2-3 percent error in the reconstructed image based solely upon the decomposition and reconstruction method. A wavelet-based approach is shown to reduce the error to effectively zero. In particular, the Daubechies orthogonal wavelet set, which was used in this research, had a reconstruction error of 10^{-9} . This means the largest difference between the pixel values in the original image and the pixel values of the decomposed and reconstructed image was on the order of 10^{-9} .

A final question is how does the method presented in this chapter combine contrast sensitivity and wavelets to form a multiresolution fusion algorithm? The answer is that it uses a wavelet-based approach to form the multiresolution pyramid and then it uses the CSF of the human visual system to analyze the details of the multiresolution pyramids. That analysis determines what combination of details should be preserved in the composite image. The premise is that the detail coefficients in a Daubechies decomposition can be treated as an image and that the CSF of the human visual system can be used to weight those details. The weighting provides a saliency value that can be used to determine the appropriate combination of the details to form the composite image. The following section provides a more detailed explanation of the algorithm.

3.2 Fusion Algorithm

This section describes the physiologically motivated image fusion algorithm developed for this research. It is a hierarchical image fusion algorithm that is performed in three stages: 1) Decomposition, 2) Fusion, and 3) Reconstruction, each of which is discussed below.

3.2.1 Stage I: Wavelet Decomposition. During the decomposition stage, each input image is decomposed into a set of detail coefficients and an approximation. The wavelet decomposition and reconstruction method is described in the background section on Wavelets. A Daubechies 20-tap orthogonal wavelet set is used to perform the Wavelet decomposition of each image to be fused. This decomposition forms a multiresolution detail pyramid where each level represents a set of wavelet coefficients at a given resolution, see Figure 3.1. Looking at Figure 3.1, the four layers at each level represent the three sets of oriented detail coeffi-

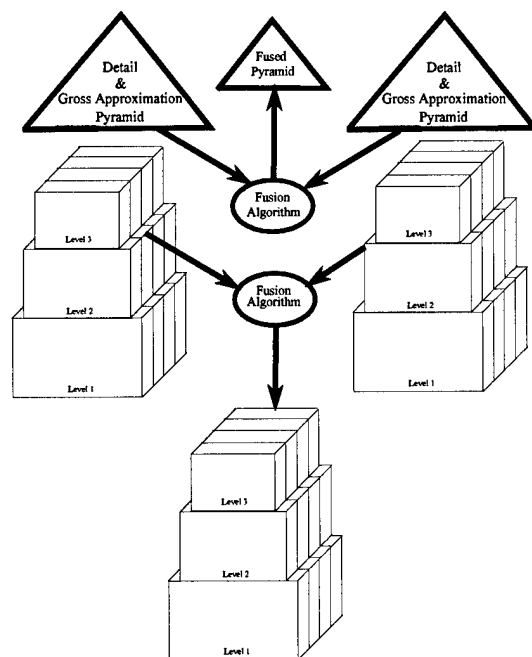


Figure 3.1 Wavelet based fusion pyramid algorithm using two input images.

cients and a set of approximation coefficients that result from a Daubechies decomposition. The number of levels of decomposition depends on the level of resolution that is desired to compare the images to be fused. In this research, the levels were limited to 6 or less. After the images are decomposed into the detail and approximation pyramids, they are ready for the Fusion Stage.

3.2.2 Stage II: Fusion. This stage of the algorithm is where the main difference lies between the method presented here and other fusion methods. All of the algorithms reviewed [1, 2, 12, 93, 94], except for Wilson's [96], based their decision to fuse on some sort of energy calculation that did not take the frequency response of the human visual system into account. The method presented here uses a weighted energy in the human perceptual domain; where the perceptual domain is based upon the frequency response (i.e., contrast sensitivity) of the human visual system.

The idea is that the human visual system responds differently to certain spatial frequencies than it does to others. Therefore, when evaluating whether two images would be perceived as different and deciding which details are more important, the evaluation needs

to be based on the perceptual abilities of the human observer. Thus, the contrast sensitivity response of the human analyst is used as the criteria to decide which parts of the input images will be averaged to form the composite image and which ones will not.

The fusion stage is performed in the following manner:

1. A saliency value S_i is computed for each of the detail and approximation coefficients from every image pyramid, see Algorithm 1.
2. The weights that determine how the detail and approximation coefficients will be combined in the composite image pyramid is calculated by using a ratio of the saliency values for each image pyramid to the sum of the saliency values over all of the input images, see Equation(3.1). This type of weighting ensures that images which have high saliency values (i.e., what an analyst would consider important) receive a large influence in the fused result and that the sum of the weights is 1.
3. The image pyramid representing the fused detail and approximation pyramid is generated by using the weights calculated in the previous step to perform a weighted sum of the input detail and approximation pyramids.

Saliency is computed as the amount of perceptual energy in a given set of detail and approximation coefficients, as related to the contrast sensitivity frequency response C of the human analyst, see Figure 3.2 and Algorithm 1.

Algorithm 1 defines $S_{i,k,l}$ as the saliency computed for a neighborhood from level l and layer k of the detail and approximation pyramid i . The level l indicates a different scale in the detail and approximation pyramid. The layer k indicates a different detail orientation or an approximation. In the Daubechies decomposition, there are 3 sets of detail orientations (vertical, horizontal, and diagonal) and an approximation. Therefore, there are 4 layers and k can be 1 – 4. The weight matrix C is the contrast sensitivity weight matrix representing the experimentally derived contrast sensitivity function (CSF) of the human visual system developed by Mannos and Sakrison [50] (Figure 3.2). The symbol F_e represents the magnitude of the energy normalized low frequency 2-dimensional Fourier components from some neighborhood at level l and layer k of the detail and approximation pyramid. The

Algorithm 1 Saliency Calculations S_i for Each Image Pyramid

Matrix Definitions $DAP_i \triangleq$ Detail and Approximation Pyramid for Input Image i $DAP_{i,l,k} \triangleq$ Detail or Approximation Coefficient Matrix at Level l and Layer k of DAP_i $\mathcal{F} \triangleq 40 \times 40$ matrix $\mathcal{F}_e \triangleq$ Energy normalized \mathcal{F} matrix $C \triangleq 40 \times 40$ Contrast sensitivity weight matrix $S_{i,l,k} \triangleq$ Saliency Matrix at Level l and Layer k for input image i **Operator Definitions** $mfft2(\cdot) \triangleq$ magnitude of the coefficients of the 2-D Discrete Fourier Transform $\odot \triangleq$ Hadamard Product (component-wise multiplication)**Constant Definitions** $cw \triangleq$ constant $I \leftarrow$ Number of Input Images $L \leftarrow$ Number of Decomposition Levels $K \leftarrow$ Number of Layers $\{L = 4; \text{for Daubechies Wavelet}\}$ $\Delta_w \leftarrow 40$ {Size of sliding window. The same as the size of C .} $\Delta_s \leftarrow 40$ {Step size to shift sliding window.}

```
for  $i \leftarrow 1 : I$  do
  for  $l \leftarrow 1 : L$  do
    for  $k \leftarrow 1 : K$  do
       $X \leftarrow \text{numrows}(DAP_{i,l,k})$ 
       $Y \leftarrow \text{numcols}(DAP_{i,l,k})$ 
       $S_{i,l,k} \leftarrow 0_{X,Y}$  { $0_{X,Y}$  zero matrix with  $X$  rows and  $Y$  columns}
      for  $x \leftarrow 1 : \Delta_s : X - \Delta_w$  do
        for  $y \leftarrow 1 : \Delta_s : Y - \Delta_w$  do
           $\mathcal{F} \leftarrow mfft2(DAP_{i,l,k}(x : x + \Delta_w, y : y + \Delta_w))$ 
           $\mathcal{F}_e \leftarrow \frac{\mathcal{F}}{\|\mathcal{F}\|}$ 
           $cw \leftarrow \text{sum}(\text{sum}(C \odot \mathcal{F}_e))$ 
           $S_{i,l,k}(x : x + \Delta_w, y : y + \Delta_w) \leftarrow cw$ 
        end for
      end for
    end for
  end for
end for
```

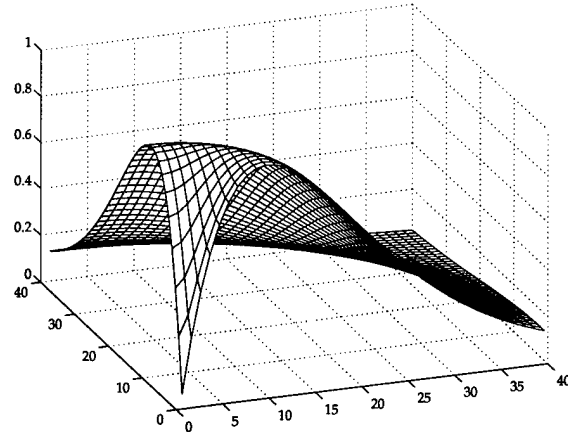


Figure 3.2 Contrast sensitivity weight matrix C representing the experimentally derived contrast sensitivity function (CSF) of the human visual system developed by Mannos and Sakrison [50].

window, which is used in the analysis of the detail and approximation pyramids, represents two degrees of visual acuity and is explained below.

The method for computing the Saliency S_i for each image detail and approximation pyramid as shown in Algorithm 1 is discussed next. The first step in computing the saliency is to energy normalize the image detail and approximation pyramids. Energy normalization is accomplished a layer at a time for all levels of the detail and approximation pyramids by dividing each coefficient (of a given layer) by the square root of the sum of the squares of all the coefficients of that layer.

Once the detail and approximation pyramids have been energy normalized, the magnitude of the low frequency 2-dimensional Fourier components for a given neighborhood (which is defined by the 40×40 window at level l and layer k of the detail and approximation pyramid are calculated. To extract the Fourier coefficients, a 40×40 window is passed over each image pyramid layer k , using a step size of 40, and the Discrete 2-D Fourier transform is computed for each resulting 40×40 set of values.

The size of the window and the step size may vary. A 40×40 window and a step size of 40 was used in this research. A window size of 40×40 was chosen because the image fusion algorithm was designed to optimally fuse images for analysis on a CRT with a resolution of 1024×768 and an average viewing distance of 24 inches from the screen. Therefore, assuming

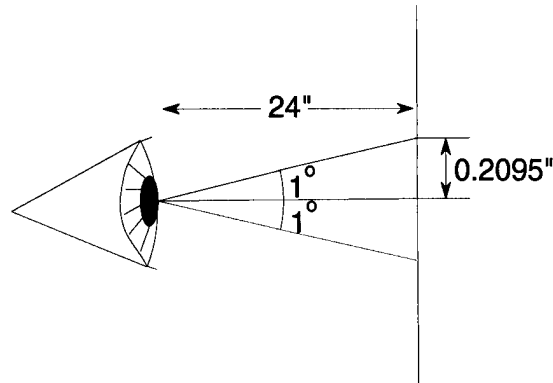


Figure 3.3 Diagram of how distance from CRT relates to Spatial Frequencies.

that a person is viewing an image on a computer screen 24 inches away, one degree of visual angle would relate to a physical viewing radius of:

$$R = D \times \tan(\theta) = 24 \times \tan(1^\circ) \text{ 0.2095 inches,}$$

where R is the viewing radius, θ is the desired visual angle, and 24 is the distance in inches from the CRT. Figure 3.3 provides an example of how the viewing angle relates to distance. Because contrast sensitivity is a function of spatial frequency, which is defined as the number of cycles per one degree of visual angle, a viewing radius of 0.2095 inches which equates to 40 pixels at a resolution of 1024×768 is desired. Therefore, a 40×40 window is chosen.

A smaller step size of 20 was tried, but the extra processing time did not produce significant improvements in noise reduction or image fusion quality.

Once the Fourier coefficients have been extracted, the non-zero frequency components are energy normalized by dividing each coefficient by the square root of the sum of the squares of the non-zero frequency coefficients. The zero frequency component values are not used. This type of normalization allows images to be compared in the same manner as a photo analyst would. That is, the photo analyst's visual system automatically discounts the illuminant in images that have large differences in average intensity, but are of the same scene. They are perceived as equivalent. Figure 3.4 is an example of two images that are identical except for some average intensity offset.



Figure 3.4 Images with different DC bias.

After the saliency values $S_{i,k,l}$ are computed, the weights $W_{i,k,l}$ for each image pyramid are computed using the following formula:

$$W_{i,k,l} = S_{i,k,l} \oslash \sum_{i=1}^I S_{i,k,l}, \quad (3.1)$$

where I is the number of images to be fused, and \oslash represents a component-wise division of the matrices.

The fused detail pyramid ($F_{k,l}$) is created by summing the weighted details from all the input images, Figure 3.1 provides a pictorial example of fusing two images. The weighted sum is defined as follows:

$$F_{k,l} = \sum_{i=1}^I W_{i,k,l} \odot D_{i,k,l} \quad (3.2)$$

Here $F_{k,l}$ is the resultant fused detail for layer k and level l , I is the number of input images, and \odot represents the Hadamard product (component-wise multiplication of the matrices). The weighting is performed by a point-by-point multiplication of the weight matrix and the detail matrix.

Once the approximation and detail pyramids have been fused into a single pyramid, the reconstruction stage is performed.

3.2.3 Stage III: Wavelet Reconstruction. This stage is simply recovering the fused image by performing a wavelet reconstruction on the composite multiresolution image pyramid formed in the fusion stage. The procedure is described in the background section on Wavelets.

3.3 Experiments

In order to evaluate the capabilities of the different fusion algorithms [2, 12, 93, 94, 96], multiple test images are generated with varying signal-to-noise ratios (SNRs) and types of backgrounds. The SNRs in decibels (db) were computed as follows:

$$SNR = 10 \log_{10} \left(\frac{Energy_{signal}}{Energy_{noise}} \right), \quad (3.3)$$

where $Energy_{signal}$ is the sum of the squared pixel grey-scale values in the pure unmodified images, $Energy_{noise}$ is the sum of the squares of the random noise. To create the test images, noise with a Uniform distribution on the interval $[0.0, 1.0]$, is scaled by a constant and is then added to various locations in the images. Two different base images are used to obtain multiple types of backgrounds. The first image is the original version of Lenna [63] which contains areas of high frequencies, vertical/horizontal edges, low frequencies, and circular oriented details, left image in Figure 3.4. The second image is the image representation of band 30 from an AVIRIS data set acquired at Moffett Field, CA (see Figure 3.5). The Moffett field image was chosen because it contains natural scenes (water, trees, fields) and urban structure (streets, runway, buildings). Band 30 was chosen arbitrarily.

Correlated or uncorrelated noise was added to the test images in three different regions (target types), thus creating sets of three images to be fused. The correlated noise was generated by convolving the uncorrelated noise with a low-pass filter. The level of correlation was computed as the full-width-half-max of the maximum amplitude of the autocorrelation of the low-pass filter [21]. The full-width-half-max value used here was 4 pixels. Figure 3.6

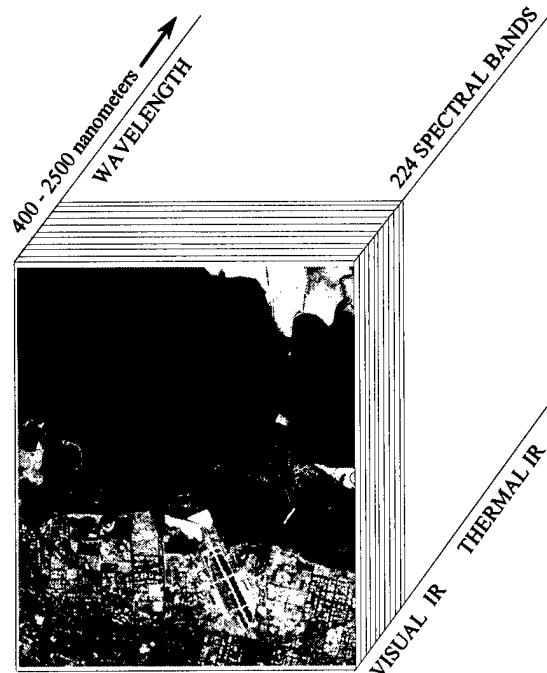


Figure 3.5 Hyperspectral Image Cube Representation taken from an AVIRIS Sensor, over Moffett Field, CA.

provides an example of three images containing uncorrelated noise with a SNR of 5db, within the area the noise was added. Figure 3.7 provides an example of three images containing correlated noise with a SNR of 5db, within the area the noise was added. Table 3.1 provides a listing of the SNRs for all of the test images using Lenna. Table 3.2 provides a listing of the SNRs for the test images using Band30. The tables include a column for the overall image SNR to provide for a comparison to the fused image results, since there will be reconstruction error to consider.

3.4 Results

In this section, the results of fusing the test images using various fusion methods are presented. The test images were fused using the algorithms by Burt and Kolczynski [12], Toet [2,93,94], Wilson [96], and the method proposed in this research.

Although, many test images were fused and analyzed, to save space, only three sets of results are presented here for comparison. One shows the results of fusion using uncorrelated noise and the other two using correlated noise and different backgrounds. Figure 3.8



Figure 3.6 Three Test Images of Lenna with 5db energy SNR of uncorrelated noise added at three different locations.



Figure 3.7 Three Test Images of Lenna with 5db energy SNR of correlated noise added at three different locations.

Image Name	Type Noise	SNR of Affected Areas (db)	SNR of Overall Image (db)
Lenna_ 10db1	Uncorrelated	10	21.9
Lenna_ 10db2	Uncorrelated	10	23.9
Lenna_ 10db3	Uncorrelated	10	20.0
Lenna_ 5db1	Uncorrelated	5	16.9
Lenna_ 5db2	Uncorrelated	5	18.9
Lenna_ 5db3	Uncorrelated	5	15.0
Lenna_ 2.5db1	Uncorrelated	2.5	14.4
Lenna_ 2.5db2	Uncorrelated	2.5	16.3
Lenna_ 2.5db3	Uncorrelated	2.5	12.5
Lenna_ 5dbcorr1	correlated	5	17.1
Lenna_ 5dbcorr2	correlated	5	18.9
Lenna_ 5dbcorr3	correlated	5	15.1

Table 3.1 Signal-to-Noise Ratios for the Lenna Modified Test Images.

Image Name	Type Noise	SNR of Affected Areas (db)	SNR of Overall Image (db)
band30_ 5dbcorr1	correlated	5	21.1
band30_ 5dbcorr2	correlated	5	19.8
band30_ 5dbcorr3	correlated	5	15.9

Table 3.2 Signal-to-Noise Ratios for the Band 30 Modified Test Images.

Image Method	Image Group	SNR of Overall Image (db)
Toet	Lenna_ 10db	14.6
Burt and Kolczynski	Lenna_ 10db	20.0
Non-Wavelet Contrast Sensitivity	Lenna_ 10db	24.2
Wavelet Contrast Sensitivity	Lenna_ 10db	26.4
Toet	Lenna_ 5db	9.5
Burt and Kolczynski	Lenna_ 5db	14.5
Non-Wavelet Contrast Sensitivity	Lenna_ 5db	19.6
Wavelet Contrast Sensitivity	Lenna_ 5db	21.3
Toet	Lenna_ 2.5db	6.9
Burt and Kolczynski	Lenna_ 2.5db	11.7
Non-Wavelet Contrast Sensitivity	Lenna_ 2.5db	17.1
Wavelet Contrast Sensitivity	Lenna_ 2.5db	18.7

Table 3.3 Signal-to-Noise Ratios for the Fused Test Images Containing Uncorrelated Noise.

Image Method	Image Group	SNR of Overall Image (db)
Toet	Lenna_ 10dbcorr	9.9
Burt and Kolczynski	Lenna_ 10dbcorr	15.5
Non-Wavelet Contrast Sensitivity	Lenna_ 10dbcorr	19.6
Wavelet Contrast Sensitivity	Lenna_ 10dbcorr	21.5
Toet	band30_ 5dbcorr	11.2
Burt and Kolczynski	band30_ 5dbcorr	16.0
Non-Wavelet Contrast Sensitivity	band30_ 5dbcorr	20.0
Wavelet Contrast Sensitivity	band30+correlated noise	22.8

Table 3.4 Signal-to-Noise Ratios for the Fused Test Images Containing Correlated Noise.

represents the results of fusing the first three images defined in Table 3.1. Figure 3.9 represents the results of fusing the three images defined in Table 3.1 that use correlated noise. Figure 3.10 represents the results of fusing the three images defined in Table 3.2 that use correlated noise.

All of the image fusion reported in this research was implemented in the following manner:

Burt and Kolczynski's fusion algorithm [12] was implemented using the following parameters:

1. Six layers of decomposition using Burt and Kolczynski's recommended w matrix and $d1$ thru $d4$ filters.
2. A 3×3 p matrix of all ones.



(a) Toet



(b) Burt and Kolczynski



(c) Non-Wavelet Contrast Sensitivity



(d) Wavelet-Based Contrast Sensitivity

Figure 3.8 Fusion results of test images using uncorrelated noise added to Lenna. The test images were the first three images defined in Table 3.1



(a) Toet



(b) Burt and Kolczynski



(c) Non-Wavelet Contrast Sensitivity



(d) Wavelet-Based Contrast Sensitivity

Figure 3.9 Fusion results of test images using uncorrelated noise added to Lenna. The test images are the images defined in Table 3.1 that contain correlated noise.



(a) Toet



(b) Burt and Kolczynski



(c) Non-Wavelet Contrast Sensitivity



(d) Wavelet-Based Contrast Sensitivity

Figure 3.10 Fusion results of test images using correlated noise added to Band30. The input images are the three images defined in Table 3.2 that contain correlated noise.

3. An alpha of 0.9.

Toet's fusion algorithm [2, 93, 94] was implemented using six layers of decomposition and Toet's recommended weight matrix w .

The Non-wavelet based contrast sensitivity fusion algorithm [96] was implemented using:

1. Six layers of decomposition using Burt and Kolczynski's recommended w matrix [12].
2. A window size of 40×40 .
3. A shift of 8.
4. A threshold of 0.20.

Again, referring to Figure 3.1, the number of layers relate to the levels in the multiresolution analysis pyramid and the window size and shift have similar relationships to the window size and shift used in the method presented here.

The Wavelet based contrast sensitivity fusion was implemented using:

1. A Daubechies 20-tap orthogonal wavelet set with one to four layers of decomposition.
2. A window size of 40×40 .
3. A shift of 40.

Three composite images were created for visual reference to aid in analysis of the fusion results, Figures 3.11, 3.12, 3.13. The composite images were created by inserting the noise from each input image into their respective locations in the original image of Lenna or Band 30, respectively.

It can be seen by comparing the fused images in Figure 3.8 with the reference images of Figure 3.11, and by looking at the fusion results in Table 3.3 that the method developed in this research does a better job of de-emphasizing uncorrelated noise in the input images than either Burt's or Toet's methods or the non-wavelet based approach. It can also be seen by comparing the fused images in Figures 3.9 and 3.10 with the reference images of Figures 3.12



Figure 3.11 Original image of Lenna and a composite image where each area of noise contains a 10 db SNR of uncorrelated noise added.

and 3.13 and by looking at the fusion results in Table 3.4 that the method developed here also does a better job of de-emphasizing correlated noise in the input images.

In every case, the method presented here produced a fused image that had a better SNR than any of the individual input images. It is important to note however, that SNR alone is not always significant when the image will be viewed by a human analyst. How the human perceives the image is ultimately the true test. For example, two images that have different SNRs, may not be “perceived” as different [34, 61, 84]. Thus, in every result reported in this dissertation, the images were also compared (informally) by human analysis to determine if better SNRs also meant better “perceived” images. In every case, the answer was yes.

3.5 *Fusion of AVIRIS Hyperspectral Image Data*

3.5.1 Fusion of IR and Visible Bands of AVIRIS Image Data. In this section, the results of fusing three bands of image data from the AVIRIS hyperspectral sensor are presented as a representative example of the algorithm. The three bands include one image from the visible frequency range and two from the infrared. Specifically, they are band numbers 30, 60, and 90, which are in the $0.67\mu\text{m}$ - $.68\mu\text{m}$, the $0.94\mu\text{m}$ - $.95\mu\text{m}$, and the



Figure 3.12 Original image of Lenna and a composite image where each area of noise contains a 10 db SNR of correlated noise added.

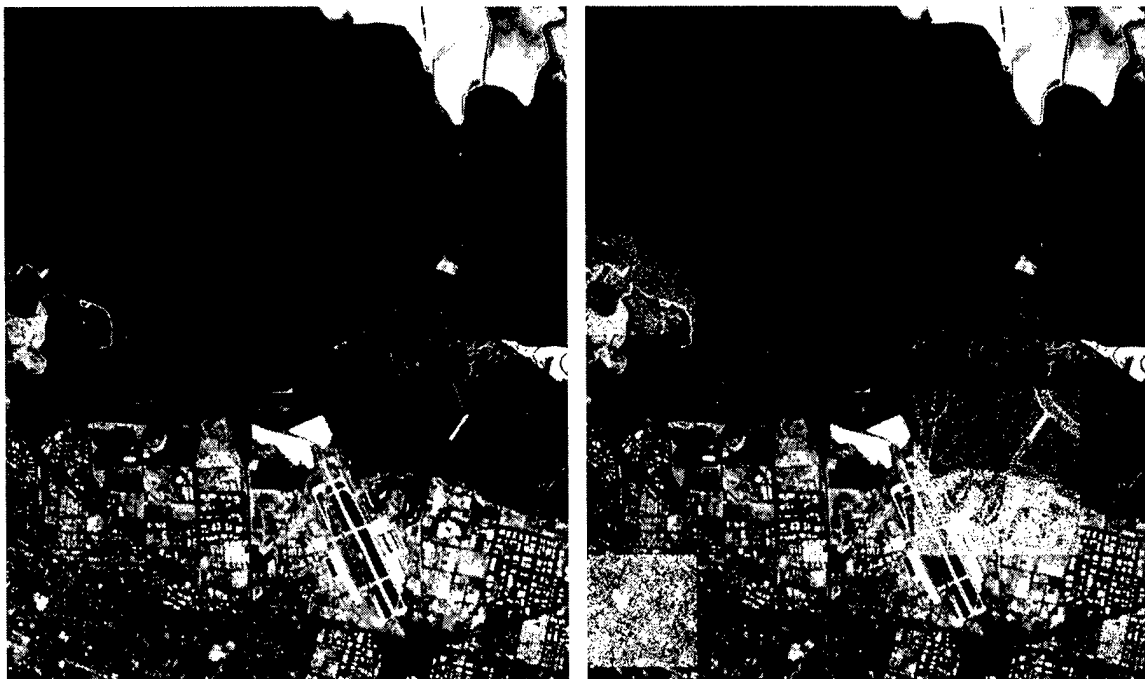


Figure 3.13 Original image of Band 30 and a composite image where each area of noise contains a 10 db SNR of correlated noise added.

1.22 μm - 1.23 μm band-pass ranges, respectively. Again, fusion was performed using the four fusion methods previously described with the same parameters as before. Figure 3.14 represents the three AVIRIS input images.

It can be seen by comparing the fused images in Figure 3.15 with the input images in Figure 3.14 that all four methods do a good job of preserving visual information from each input image. However, comparing the four fused results to each other, Toet's fusion method does not have the same amount of detail that is present in the other three. For example, looking at the upper right hand corner of the fused images, it is easy to see that Toet's method causes details to become washed out. In order to make the details in the upper corner more distinguishable, the overall intensity level of the image has to be reduced. The reduction in intensity then causes the other areas of the image to become less distinct. Also the runway and surrounding area is more clear in Burt and Kolczynski's method and the contrast sensitivity methods than it is in Toet's fusion method.

Comparing the fusion results of the contrast sensitivity methods with Burt and Kolczynski's shows that they all have similar characteristics. Each method appears to preserve features in the input images that are dominant. An example of this preservation is observed by looking at the road that is a dominant feature in bands 60 and 90 but not in band 30. The road is located in the lower third of the image and extends from the left edge of the image all the way to the right. It is clearly seen as a continuous road in bands 60 and 90, but is hard to distinguish in band 30. Looking at the fused results in figure 3.15 the road is preserved in the composite. Overall, the contrast sensitivity methods provide better results, both aesthetically and numerically (SNR), to either Burt and Kolczynski's or Toet's methods.

Finally, comparing the fusion results of the non-wavelet versus the wavelet based contrast sensitivity methods, they are practically identical, Figure 3.15. This was expected since the method of determining the salient features to retain in the composite image is the same (i.e., contrast sensitivity). The advantage would appear in a possible overall image improvement due to a reduction in reconstruction error. This was evident in the previous section when SNRs were compared. Here, the differences are hard to detect. A final example



(a) Band 30



(b) Band 60



(c) Band 90

Figure 3.14 This figure represents three bands from the AVIRIS hyperspectral sensor.



(a) Toet



(b) Burt and Kolczynski



(c) Non-Wavelet Contrast Sensitivity



(d) Wavelet-Based Contrast Sensitivity

Figure 3.15 This figure represents the fusion of AVIRIS bands 30, 60, and 90.



Figure 3.16 Wavelet-Based Contrast Sensitivity Fusion Results of AVIRIS Bands 30 to 40 is presented in the next section that demonstrates how well this new fusion method can fuse many bands from a hyperspectral sensor.

3.5.2 Fusion of Sets of Hyperspectral AVIRIS Image Data. The examples in this section are provided as a demonstration that the fusion method developed in this article can also fuse many spectral bands from the AVIRIS hyperspectral sensor without causing loss of information or dynamic range. The example in the next chapter which combines fusion and RICA to perform ATC will demonstrate numerically how information is maintained and even enhanced through fusion. The following figure is intended as a visual example of how the dynamic range is preserved. Figure 3.16 represents the results of fusing bands bands 30 through 40 from the AVIRIS hyperspectral sensor. The combined band-pass range of the images is from $0.67\mu\text{m}$ - $.75\mu\text{m}$.

3.6 Conclusions

Image fusion using contrast sensitivity as a salience measure produced fused images that were visually better than either Burt and Kolczynski's [12] or Toet's [2,93,94] methods.

It is also shown in Table 3.3 and Table 3.4 that contrast sensitivity based fusion, in conjunction with a wavelet-based decomposition and reconstruction, results in better SNRs over a wide range of noise types, levels, and backgrounds. Using a paired t -test and the results of Tables 3.3 and 3.4, this new fusion method statistically outperforms the other fusion algorithms by an average of 6.8db in SNRs. Also, as the noise in the images was increased, the performance of the contrast sensitivity based fusion actually improved over the other methods. In other words, the differences in the SNRs between the physiologically motivated fusion method and the others was greater as the noise in the input images was increased. For example, as the noise was increased and the SNR of the input images went from 10db to 2.5db, the relative difference between Burt and Kolczynski's method and the new fusion method went from 13.8 percent to 23.0 percent.

The non-wavelet based contrast sensitivity had excellent results and improved the SNRs in most cases. However, there were instances where the fused image did not have a better SNR than some of the input images. This occurred with at least one example of the band 30 data from the AVIRIS sensor. This is one of the reasons the wavelet based approach was developed. The SNR was affected by the reconstruction error. It has been shown that the wavelet based approach removes this problem and in all cases the fused image had a better SNR than any of the input images. Thus, contrast sensitivity based multiresolution image fusion using a wavelet basis set is an excellent method of image fusion for hyperspectral image data.

Contrast sensitivity based fusion can also fuse image data from the AVIRIS hyperspectral sensor and still maintain the relevant visual information in the input scenes. Figures 3.15 and 3.16 clearly demonstrate that images from multiple spectral bands can be fused to reduce image storage and increase the amount of information in a given scene. Additionally, the results in Section 4.6 show that fusion of hyperspectral data not only preserves information it enhances it.

At this point, it is important to reiterate why the image fusion algorithm was developed. It was developed to reduce the vast amounts of hyperspectral image data to a smaller subset of images that could then be fed to a recognizer. When the recognizer is a human analyst,

reducing the number of images they have to analyze, while increasing image detail, reduces the amount of time spent analyzing images. If the recognizer is an automated system, such as an ANN, reducing the dimension (i.e., Cardinality) of the data set is important in preventing the "Curse of Dimensionality" [7], which was discussed in the background chapter. In either case, image fusion is a critical first step in the process of target detection in hyperspectral image data and the physiologically motivated fusion algorithm presented is superior to others at that task. The next chapter describes the self-architecting Radial Basis Function Iterative Construction Algorithm (RICA) that was designed to detect targets in the fused hyperspectral image data.

IV. Radial Basis Function Iterative Construction Algorithm (RICA)

The purpose of this chapter is to describe the methods that RICA uses to select and train the hidden layer nodes and to optimize the output layer weights. Also, experimental cases are presented which show that RICA outperforms several popular fixed and constructive types of ANN architectures. In addition to empirical evidence, there is a mathematical proof presented which shows that RICA performs a minimum mean squared-error approximation to a Bayes optimal discriminant function. Finally, an example is presented showing how RICA works in conjunction with the physiologically motivated fusion algorithm discussed in Chapter III to perform ATC of hyperspectral image data.

4.1 Selecting and Training The Hidden Layer Nodes

RICA uses a two stage process to select and train the hidden layer nodes. The first stage autonomously determines the number of hidden nodes needed to represent the probability density function (pdf) for each class, based upon the input data. The second stage calculates the values needed to parameterize each hidden node (i.e., the mean and covariance of each Gaussian basis function).

During first stage training, RICA uses a combination of Mahalanobis distance clustering and a Gaussian Goodness-of-Fit measure [9,48,86,87] to determine how many nodes are needed to represent the pdf of each class. Obviously not all pdfs are Gaussian, but as mentioned in Section 2.3.1, RBFs using Gaussian basis functions are Universal Function Approximators and that all Lebesgue p -integrable (L^p) functions can be approximated by a linear combination of Gaussian basis functions [9,30,59,64]. Therefore, the problem becomes one of finding a sufficient number of basis functions (hidden nodes), their parameters, and the proper linear combination of those basis functions.

The first step RICA uses to determine the number of hidden layer nodes is to separate the data using the class labels. Separation by class allows RICA to use the information in the labels to better approximate the pdf of each class separately. This is a different approach than most other methods which use all of the data, regardless of class, to perform probability

density estimation. Section 2.3.1.2 describes several of those techniques and their limitations. The argument for separating the data is that we are trying to estimate the pdf of each class separately and the distribution of each class is independent of the other classes, there is no need to mix the classes. Combining the classes only serves to add confusion and complexity to the problem, especially since the class membership of each of the training exemplars is known *a priori*.

Once the classes have been separated, the next step is to determine how many nodes are needed to represent each of their respective pdfs. One method is to use a Parzen approach to represent the class pdfs, in which case, a Gaussian function with a fixed width would be placed at each sample point for each class [9, 60, 76]. This is the extreme case or upper bound for generating the number of functions necessary for density estimation. Another method is to use a single Gaussian function with a common covariance matrix to represent the pdf of each class, this would represent a lower bound for representing the class pdfs. Neither extreme is likely to result in good network performance over a wide field of data sets. If too many basis functions are used, the network does not tend to generalize well. If too few basis functions are used, the network will not have the representational power to adequately approximate the discriminant function and classification accuracy will be poor. One resolution to this problem is to use a human operator to preselect some number of nodes and then use clustering (*K*-means, Euclidean, LVQ, etc.) or Expectation Maximization to determine the parameters of the Gaussian functions [9]. The goal of this work was to find an appropriate number of Gaussian functions without using one at every sample point and without using human input.

Because Gaussian functions are being used as the hidden node basis functions and there was a desire to not place a basis function at every sample point, a method was needed to find groups of data points that could be reasonably modeled by a single Gaussian. The problem is two-fold. The first is given a subset of points from a set of data, is there a method to determine if the points can be reasonably approximated by a Gaussian function. The second is deciding which subset of points should be chosen. Some density estimation methods use either a Chi-Squared test or a Kolmogorov-Smirnov test to decide if the subset

of points can be reasonably approximated by a Gaussian function [18]. However, D'Agostino states in his book on Goodness-of-fit techniques that the Chi-Squared test, in general, is not a powerful test of normality. He goes on to say that the Kolmogorov-Smirnov test is only an historical curiosity and should never be used. RICA uses the Shapiro-Wilk goodness-of-fit test [18, 86, 87], which is presented in more detail in Section 4.1.2, to determine if a subset of points can be reasonably modeled by a single Gaussian function. Not only has the Shapiro-Wilk test been shown to be an excellent test for this type of problem [18, 86, 87], D'Agostino says that it is a more powerful test of normality than either the Chi-Squared or Kolmogorov-Smirnov tests [18].

The Shapiro-Wilk test analyzes a set of points and outputs a number between 0 and 1 that represents a confidence value that the set of data came from a Gaussian distribution. The larger the value the more confidence there is that a particular subset of points came from (and thus can be represented by) a Gaussian function. Representing a subset of points by a single Gaussian function instead of representing each point with a Gaussian function greatly reduces the number of nodes needed. For example, given a set of 100 points that came from a mixture of two univariate Gaussians with means 0 and 1 and variances 0.1 and 0.1 respectively, the Parzen approach would use 100 Gaussian functions to represent the pdf. In reality, two functions with means 0 and 1 and variances 0.1 and 0.1 would be sufficient to represent the pdfs. An added benefit is that reducing the number of basis functions and subsequently the number of free parameters in the network should improve the generalization capability of the network.

While it has been shown that the Shapiro-Wilk test is an appropriate test to decide if a group of points can be reasonably represented by a single Gaussian function, it does not solve the problem of choosing which subsets of points in each class need to be grouped together. RICA uses a combination of the Shapiro-Wilk test, Mahalanobis Distance clustering, and *K*-means to determine which subsets of points of each class can be represented by separate Gaussian functions. An overall description of the algorithm is discussed and then the step-by-step details are presented below. Section 4.1.3 discusses the Mahalanobis distance clustering algorithm.

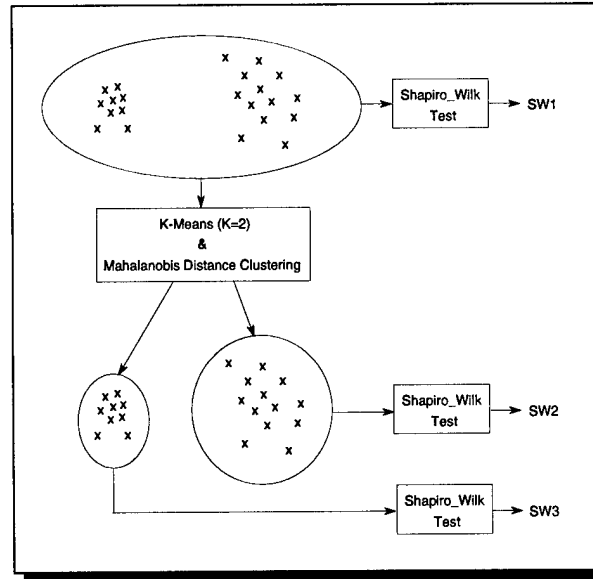


Figure 4.1 This Figure illustrates the process involved in splitting the data set into two parts and measuring the Shapiro-Wilk test on the original cluster and then the two new clusters.

RICA determines how many hidden nodes are needed to represent each class conditional pdf by first separating the data by class using the class labels. Next, RICA partitions each set of class data into smaller and smaller partitions until no further improvement in the Shapiro-Wilk confidence value is obtained (see Figure 4.1). This is accomplished by examining each partition to determine if a higher Shapiro-Wilk value could be obtained by splitting the partition into two new partitions. If a split occurs, the new partitions will be examined to determine if they should be split. The evaluation and possible splitting continues until no further improvement in the Shapiro-Wilk value is obtained. Once the partitioning is finished for each class, the unbiased estimators of the sample mean and covariance of each partition are calculated [32] and those values are used to define the center vector and covariance of the Gaussian functions used in the hidden layer nodes. Once the hidden layer node basis functions have been calculated, the best linear combination of those basis functions is determined during the output layer weight training phase which is discussed in the next section. Now that a general description of RICA has been presented the actual mechanics follow:

4.1.1 Algorithm for Selecting and Training The HLN's. Separate the data by class label and then perform the following steps on each of the classes separately. It is important

to note that the following steps are iterative and that the process starts with all of the training data in each individual class and is iteratively performed on each new cluster as it is formed. The goal is to analyze each cluster to determine if it can be well represented by a single Gaussian or if it should be broken into smaller clusters, each of them represented by a single Gaussian.

Algorithm to determine Hidden Layer Nodes

1. Repeat until all clusters have been analyzed. (i.e., The process is then repeated for each "new" cluster until no more improvement is found (i.e., additional splitting of the data does not result in a larger Shapiro-Wilk value), see Figure 4.1.
2. Let D_1 denote the data in the cluster under test. In the beginning of each class under analysis, the data D_1 will be the whole data set for a given class. After the first step, the data D_1 will represent each new cluster that has been generated in the following steps.
3. Let $SW1$ be the Shapiro-Wilk confidence value for data D_1 using the algorithm presented in Section 4.1.2, see Figure 4.1.
4. Separate the data D_1 into two clusters using the Mahalanobis distance clustering algorithm presented in Section 4.1.3. Let D_2 and D_3 represent the two new clusters formed from D_1 .
5. Let $SW2$ and $SW3$ be the Shapiro-Wilk confidence values for data D_2 and D_3 using the algorithm presented in Section 4.1.2, see Figure 4.1.
6. If $SW2$ or $SW3$ is greater than $SW1$, retain the new clusters D_2 and D_3 and remove the super-cluster D_1 . If $SW1$ is greater than $SW2$ and $SW3$, then retain the super-cluster D_1 .
7. Goto Step 1

4.1.2 Shapiro-Wilk Goodness-of-Fit Algorithm. The Shapiro-Wilk test is a test of Normality based upon the ratio of estimates of scale [9, 18, 48, 86, 87]. It is the ratio of a

squared sum of weighted order statistics to the squared sample variance times the number of samples, given by

$$W = \frac{\left(a_i X_{(i)}\right)^2}{S^2}, \quad (4.1)$$

where W is the Shapiro-Wilk value, $X_{(i)}$ are the ordered samples, a_i are the weighting coefficients, S^2 is N times the squared sample variance, N is the number of samples, and $i = 1, \dots, N$. The weighting coefficients were calculated by Shapiro-Wilk such that the W ratio is between 0 and 1 and it approaches 1 as the distribution of the sample data becomes more Gaussian. A table containing the a_i coefficients can be found in the book by D'Agostino entitled "Goodness-of-fit Techniques" [18].

4.1.3 Mahalanobis Distance Clustering Algorithm. Mahalanobis distance clustering (MDC) which has been successfully used in other ANN architectures [9, 33] separates data into clusters based upon the Mahalanobis distance between a sample vector and the current cluster centers. Given the input sample x , a cluster center m , and \mathbf{C} the sample covariance matrix of the data points contained in a given cluster, the Mahalanobis distance ($D_M(x, m, \mathbf{C})$) is defined as:

$$D_M(x, m, \mathbf{C}) = (x - m)^t \mathbf{C}^{-1} (x - m),$$

Clustering is performed by placing each data vector into the cluster that has the smallest Mahalanobis distance between the data vector and the vector representing the cluster center. Once the clusters are formed, a new sample mean and covariance is computed for each cluster. The data is then reclustered using the new cluster centers and covariances. This is performed iteratively until the clusters stop changing. This can be measured either by checking the membership of each sample point and testing to see if its membership changed from one iteration to the next or by checking the cluster centers to see if they have changed.

One important problem that occurs when calculating the covariance is that the inverse covariance may not exist. This occurs if the number of data points is less than the dimension of the data. For Radial Basis Function (RBF) networks, the rule of thumb is to have at least $d + 1$ samples per basis function (d is the feature dimension) to estimate the covariance

matrix of each Gaussian basis function [20]. Several times that amount may be needed to smooth out statistical fluctuations and to get good estimates [20].

Having $d + 1$ samples does not guarantee that the inverse covariance matrix will exist. Sometimes the samples are linearly dependent along some hyperplane or line and can be projected into a subspace that has a smaller dimension than the feature space. When this occurs, the rank of the covariance matrix is less than the dimension of the data and the inverse covariance will not exist. RICA overcomes this problem by calculating the eigenvalues of the covariance matrix and if there is an eigenvalue close to zero, (the rank of the matrix is less than the dimension of the feature space), the covariance matrix is close to being non-invertible and needs to be modified. The modification is accomplished by replacing the eigenvalue or values that are close to zero with some minimum threshold value and using the eigenvectors and the new eigenvalue to create a new approximate covariance matrix.

Another problem is starting the MDC algorithm given a set of data and no centers or covariances. Since RICA uses the MDC algorithm to split a partition of data into two new partitions, the number of centers is known and is always two. The next step is to find suitable centers. The K -means algorithm, with a K value of 2, was chosen because it yields two cluster centers that are from the two most dense areas of the input partition data. This is desirable because the center of a Gaussian function is the most dense area of the function.

Once the K -means algorithm is used to find two center vectors, the Euclidean distance metric is used to initially partition the data into two clusters. Now that the data has been partitioned into two clusters, the sample mean and covariance are calculated for each. Next, the data are repartitioned using the current centers and covariances, but now the Mahalanobis distance is used. Now the algorithm performs the center calculation and partitioning iteratively, by calculating the new cluster centers and covariances until the centers no longer change.

The reason a Mahalanobis distance metric is used instead of the Euclidean metric is because clustering using the Euclidean metric tends to create clusters that are of equal size and are spherical in shape [33]. Euclidean distance clustering will also incorrectly cluster data that comes from two clusters when the between-cluster variance is larger than the within-

cluster variance [33,99]. This can cause large clusters and clusters that have elongated shapes (a cigar shape) to be broken up. Another reason the Mahalanobis distance is chosen over the Euclidean is because Gaussian functions model localized receptive fields (Gaussian functions will produce similar outputs for vectors that are close in Mahalanobis distance). This is easily seen by comparing the exponent of a Gaussian function with the previous definition of the Mahalanobis distance metric.

$$Gauss(x, m, \mathbf{C}) = \frac{1}{2\pi\sqrt{|\det \mathbf{C}|}} e^{-\frac{(x-m)^t \mathbf{C}^{-1} (x-m)}{2}},$$

where x , m , and \mathbf{C} have the same definition as for the Mahalanobis distance metric defined above.

The method presented here finds the smallest number of nodes required to maximize the confidence (Shapiro-Wilk measure) that each partition is well represented by a Gaussian function. Reducing the number of nodes in the hidden layer reduces the number of free parameters in the network which improves the generalization capability of the network. Once the hidden layer nodes have been selected and trained, the output layer nodes are trained.

4.2 Training The Output Layer

The previous section described how RICA's hidden layer nodes (HLNs) were selected and trained. This section describes how RICA's output nodes are trained. RICA uses a 1-of- M coding scheme for the output, where M is the number of classes. In a 1-of- M coding scheme, there is only 1 output node whose desired output is positive for a given input. The remaining nodes have desired outputs which are less than or equal to zero. In a 1-of- M coding scheme there is exactly 1 node per class. Therefore, in a 1-of- M coding scheme the node that produces a positive output is the one corresponding to the node that is assigned to the class of the particular input. For example, given an input training sample that is from class three, the desired output for node three would be greater than zero while all other nodes would be trained to be less than or equal to zero. RICA trains the output nodes so that the outputs are positive for the correct class and negative for the incorrect class. Unlike MLPs whose output node activation functions can be non-linear (Sigmoidal or Hyperbolic

Tangent functions, etc), RICA's output activation functions are linear. With linear outputs, the output of each node is the weighted sum of the outputs of the HLN's. The problem is to find the proper weight vector to achieve the desired outputs.

Because Gaussian units are used for the HLN's, they have localized receptive fields whose outputs are a function of the Mahalanobis distance between an input pattern and the centroid of the HLN. The outputs of the HLN's fall off rapidly to zero after the Mahalanobis distance becomes greater than three (three standard deviations from the center). Since the HLN's have been selected and trained to fit the distribution of each class separately, they will project the input data onto a linearly separable space, except where the data overlap (in a Mahalanobis distance clustering sense), in the input space. Given that the HLN's are likely to project the input data onto a linear separable space the Ho-Kashyap method [20, pp 159-166], which is discussed in detail in Section 2.3.1.3, was chosen to optimize the output layer weight vectors. Details of the Ho-Kashyap algorithm are condensed and presented below for reference.

4.2.1 Ho-Kashyap Algorithm. This section presents the Ho-Kashyap algorithm in pseudo code form. It is important to note that the Ho-Kashyap algorithm is used to train the output layer weights after the hidden layer nodes have been selected and trained.

In the Ho-Kashyap algorithm presented in Algorithm 2, iterating while $e_i^+ > 0$ will either yield an $e_i = 0$ or an $e_i < 0$. If $e_i = 0$ the data are linearly separable and we have a solution to our problem of finding a separating vector. If $e_i < 0$ we still have a MSE approximate to the separating vector, but the data are not linearly separable. A final note on Ho-Kashyap algorithm is that because the Ho-Kashyap method is used to find the weight vector for each output node separately, training the output layer weights can be implemented on a parallel architecture. This would offer a considerable speed-up in training time.

The previous sections described how RICA autonomously selects the number of hidden layer nodes needed and optimizes the parameters needed to fully describe the architecture. This was very important in showing that RICA is an autonomously adaptive ANN architecture. The next section proves that RICA is not only adaptive, but that it has the desirable

Algorithm 2 Ho-Kashyap Algorithm

$T \triangleq N \times D$ training data matrix
 $T_m \triangleq$ elements of T corresponding to a class other than m are multiplied by -1
 $Y_m^* \triangleq T_m; 1$ { T_m is augmented by a column of ones}
 $Y_m^{*\dagger} \triangleq$ Pseudoinverse of Y_m^*
 $M \leftarrow$ Number of Output Nodes (one per class)
 $N \leftarrow$ Number of training samples in T
 $D \leftarrow$ Number of features
 $i \triangleq$ Iteration number
for $m \leftarrow 1 : M$ **do**
 $i = 1$
 $e_i = 1000$ {An arbitrary value greater than zero}
 Create Y_m^*
 Choose and arbitrary $b > 0$ { b is an $N \times 1$ vector with values typically greater than 1 for this work}
 while $e_i > 0$ All elements greater than zero **do**
 $a_i \leftarrow Y_m^{*\dagger} b_i$
 $e_i \leftarrow Y_m^* a_i - b_i$
 $e_i^+ \leftarrow \frac{1}{2}(e_i + |e_i|)$ { $|e_i|$ is the vector of absolute values of the components of e_i }
 $b_{i+1} \leftarrow b_i + 2\rho e_i^+$ { $0 \leq \rho < 1$ }
 $i \leftarrow i + 1$
 end while
end for

property of being a minimum MSE approximation to a Bayes optimal discriminant function. This makes it a very powerful classification ANN.

4.3 RICA: a minimum MSE approximation to a Bayes optimal discriminant function

It will be shown that RICA performs a minimum mean squared-error approximation to a Bayes optimal discriminant function (BODF) and that RICA's outputs can be interpreted as *a posteriori* probabilities. Ruck, et al, [81] and Richard [71], and Duda and Hart [20] showed that a MLP using back-propagation and special assumptions is a mean squared-error approximator to the BODF. The proof for RICA will use a similar technique to that of Ruck et al and Richard, however, it will be shown for a RBF network and it will present a more general result than that of either Ruck, et al, or Richard. Ruck, et al, presented the proof for a MLP with a fixed architecture and assumed the existence of continuous pdfs. Richard also presented a more general multi-class proof for neural networks performing a minimum squared-error optimization. However, Richard also assumed a continuous pdf. The Theorem presented here is a generalized proof for any type of architecture that performs a minimum mean squared-error optimization and it makes no assumptions about the existence of the pdfs. That is, the cumulative distribution function may not be differentiable.

Theorem. *If a neural network architecture performs a mean squared-error minimization between the desired and the actual output using a 1-of- M coding scheme, then the neural network performs a mean squared-error approximate to the Bayes optimal discriminant function (classifier) and the outputs can be interpreted as Bayesian a posteriori probabilities.*

Before the proof is presented, a brief description of what it means to be a Bayes optimal discriminant function is needed. In most pattern recognition classification problems, the goal is to correctly classify an input pattern x into 1-of- M classes. A Bayes optimal classifier is one that calculates the Bayesian probability $P(C_m|x)$ for each class C_m and then selects the class that has the highest probability [9,20,32,71,81,85]. The Bayesian probability $P(C_m|x)$ represents the conditional probability of class C_m given the input vector x . If $P(x|C_m)$ is the conditional probability of producing x given the class is C_m , $P(C_m)$ is the *a priori* probability of the class C_m , $P(x)$ is the unconditional probability of the input vector x , and M is the

number of classes, then $P(C_m|x)$ is defined by the Bayesian formula in conjunction with the law of probability:

$$\begin{aligned} P(x)P(C_m|x) &= P(x|C_m)P(C_m) \\ P(x) &= \sum_{m=1}^M P(x|C_m)P(C_m), \end{aligned} \quad (4.2)$$

Simply put, how likely is it that a given input sample vector x came from class C_m . Choosing the class that has the greatest Bayesian probability (i.e., the most likely), provides the best possible choice. Bayes optimal discriminant functions are functions that minimize the probability of error for a given classification by using the Bayes probability as a criterion [9, 20, 32, 71, 81, 85].

Let \mathcal{X} denote a sample space. Assume there are M nonempty classes $C_1, C_2, \dots, C_M \subset \mathcal{X}$ such that

$$\mathcal{X} = \bigcup_{m=1}^M C_m \quad (4.3)$$

and $C_i \cap C_j$ is not necessarily the empty set for each $i, j = 1, 2, \dots, M$. Then an example of a Bayes optimal discriminant function and the decision rule that accompanies it is

$$g_m(x) = P(C_m|x), \quad m = 1, 2, \dots, M, \quad x \in \mathcal{X}. \quad (4.4)$$

The decision rule is: x is from class C_m if $g_m(x) > g_j(x)$ for all $j \neq m$.

Let $F_m(x, \theta)$ represent the output of the m^{th} output node and θ represent the weights of the output layer and the means and covariances of the Gaussian basis functions used in the hidden layer nodes. To show that RICA performs a minimum mean squared-error approximation of the Bayes optimal discriminant function $g_m(x)$, it will be shown that the following error criterion is minimized during RICA's training:

$$\epsilon^2(\mathcal{X}, \theta) = \sum_{m=1}^M \int_{\mathcal{X}} [F_m(x, \theta) - g_m(x)]^2 P(dx) \quad (4.5)$$

To begin the proof, a sample error function E_s will be defined as the cumulative squared difference between the desired output of the network and the actual output of the network.

However, before the sample error function can be defined a sample set $\mathbf{X} \subset \mathcal{X}$ and the desired output $d_m(x)$ must be defined.

For each $m \in \{1, 2, \dots, M\}$, let N_m be the number of samples representing each class m , and let $N = N_1 + N_2 + \dots + N_M$, be the total number of samples in the set \mathbf{X} , then for each $m = 1, 2, \dots, M$ define

$$\mathbf{X}_m = \{x_{m,N_1}, x_{m,N_2}, \dots, x_{m,N_m}\}$$

and note that $\mathbf{X} = \bigcup_{m=1}^M \mathbf{X}_m$. Because the desired output is based upon a 1-of- M coding scheme, the desired output function $d_m(x)$ is:

$$d_m(x) = \begin{cases} 1 & \text{if } x \in \mathbf{X}_m \\ 0 & \text{otherwise} \end{cases}$$

Choosing 0 and 1 as the outputs allows the final outputs of the network to be interpreted as probabilities. The outputs could be some positive or negative constant, which is useful in solving a system of linear equations when the outputs are linear, but would require further processing to interpret the outputs as probabilities. When the outputs are allowed to be positive or negative, a softmax procedure [9, pp 215] can be used to allow the outputs to be interpreted as probabilities.

Using the above desired output function $d_m(x)$ and sample set \mathbf{X} , the sample error function is defined as:

$$E_s(\mathbf{X}, \theta) = \sum_{n=1}^N \sum_{m=1}^M (F_m(x_n \theta) - d_m(x_n))^2 \quad (4.6)$$

It will be shown that RICA minimizes E_s with respect to θ and that minimizing E_s with respect to θ also minimizes $\epsilon^2(\mathcal{X}, \theta)$ which completes the proof that RICA performs a minimum mean squared-error approximation to the Bayes optimal discriminant function.

Let the number of hidden layer nodes be fixed at H and the values of the means and covariances of the hidden layer nodes be fixed as well. The number H is arbitrary and the

means and covariances can vary for each hidden layer node. If $w_{m,h}$ is defined as the weight applied to Φ_h , which is the output of the h^{th} Gaussian hidden layer node, then the output function $F_m(x_n, \theta)$ can be rewritten as:

$$F_m(x_n, \theta) = \sum_{h=1}^H w_{m,h} \cdot \Phi_h(x_n, \theta), \quad (4.7)$$

The sample error becomes:

$$E_s(\mathbf{X}, \theta) = \sum_{n=1}^N \sum_{m=1}^M \left[\left(\sum_{h=1}^H w_{m,h} \cdot \Phi_h(x_n, \theta) \right) - d_m(x_n) \right]^2 \quad (4.8)$$

It is shown in Section 4.2 the method used to train the output layer weights w_m minimizes the squared-error function seen in the sample error function 4.8 above. Therefore, RICA minimizes the sample error function $E_s(\mathbf{X}, \theta)$. Now it is necessary to show that minimizing $E_s(\mathbf{X}, \theta)$ provides the minimum mean squared-error approximation to the Bayes optimal discriminant function, which will complete the proof.

Using $E_s(\mathbf{X}, \theta)$ defined in Equation(4.8) above, the average error is

$$\begin{aligned} E_a(\mathbf{X}, \theta) &= \lim_{N \rightarrow \infty} \frac{1}{N} E_s(\mathbf{X}, \theta) \\ &= \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N \sum_{m=1}^M (F_m(x_n, \theta) - d_m(x_n))^2. \end{aligned} \quad (4.9)$$

Using the Strong Law of Large Numbers [58], the average error $E_a(\mathbf{X}, \theta)$ is the expected value of the sample error function $E_s(\mathbf{X}, \theta)$. Applying the Strong Law of Large Numbers and letting dx be defined as the infinitesimal centered at x , $dP(x) = P(dx)$, and using the following Bayesian identities:

$$\begin{aligned} P(dx) &= \sum_{m=1}^M P(dx|C_m)P(C_m) \\ P(dx|C_m)P(C_m) &= P(C_m|dx)P(dx) \\ &= g_m(x)P(dx) \end{aligned}$$

Equation(4.9) can be rewritten as:

$$\begin{aligned}
E_a(\mathcal{X}, \theta) &= \int_{\mathcal{X}} \sum_{m=1}^M (F_m(x, \theta) - d_m(x))^2 P(dx) \\
&= \sum_{m=1}^M \int_{\mathcal{X}} (F_m(x, \theta) - d_m(x))^2 P(dx) \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} 2d_m(x) F_m(x, \theta) P(dx) + \int_{\mathcal{X}} d_m^2(x) P(dx) \right]
\end{aligned}$$

Again, no assumption is made as to the differentiability of the cumulative distribution function P . Since the desired output function has values of only 1 and 0, $d_m^2(x) = d_m(x)$ thus,

$$\begin{aligned}
E_a(\mathcal{X}, \theta) &= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} 2d_m(x) F_m(x, \theta) P(dx) + \int_{\mathcal{X}} d_m(x) P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} d_m(x) (2F_m(x, \theta) - 1) P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} d_m(x) (2F_m(x, \theta) - 1) \sum_{m'=1}^M P(dx|C_{m'}) P(C_{m'}) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} \sum_{m'=1}^M d_m(x) (2F_m(x, \theta) - 1) P(dx|C_{m'}) P(C_{m'}) \right]
\end{aligned}$$

Using the fact that when $m = m'$, $d_m(x) = 1$, otherwise $d_m(x) = 0$, the last part of previous equation can be written as:

$$E_a(\mathcal{X}, \theta) = \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} (2F_m(x, \theta) - 1) P(dx|C_m) P(C_m) \right] \quad (4.10)$$

Using the previously defined Bayesian identities Equation(4.10) becomes:

$$\begin{aligned}
E_a(\mathcal{X}, \theta) &= \sum_{m=1}^M \left[\int_{\mathcal{X}} F_m^2(x, \theta) P(dx) - \int_{\mathcal{X}} (2F_m(x, \theta) - 1)g_m(x) P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} [F_m^2(x, \theta) - 2g_m(x)F_m(x, \theta) + g_m(x)] P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} [F_m^2(x, \theta) - 2g_m(x)F_m(x, \theta) + g_m^2(x) - g_m^2(x) + g_m(x)] P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} [F_m^2(x, \theta) - 2g_m(x)F_m(x, \theta) + g_m^2(x)] P(dx) + \int_{\mathcal{X}} [g_m(x)(1 - g_m(x))] P(dx) \right] \\
&= \sum_{m=1}^M \left[\int_{\mathcal{X}} [F_m(x, \theta) - g_m(x)]^2 P(dx) \right] + \sum_{m=1}^M \int_{\mathcal{X}} [g_m(x)(1 - g_m(x))] P(dx) \\
&= \epsilon^2(\mathcal{X}, \theta) + \sum_{m=1}^M \int_{\mathcal{X}} [g_m(x)(1 - g_m(x))] P(dx) \tag{4.11}
\end{aligned}$$

The summation is independent of θ , therefore minimizing $E_a(\mathcal{X}, \theta)$ also minimizes $\epsilon^2(\mathcal{X}, \theta)$.

In conclusion, it has been shown that RICA minimizes E_s with respect to θ which also minimizes E_a with respect to θ . It has also been shown that minimizing E_a with respect to θ also minimizes ϵ^2 with respect to θ . Therefore, RICA yields a network that is a minimum mean squared-error approximation to the Bayes optimal discriminant function.

A final note of this section is that because of the previous proof

$$F_m(x, \theta) \approx g_m(x) \tag{4.12}$$

thus, the outputs of RICA may be interpreted as *a posteriori* probabilities and a level of confidence may be associated with the classification of each sample. This could prove very useful in accepting/rejecting a classification or in using RICA in a hierarchical network.

4.4 Experiments

This section describes the tests that were conducted to compare the performance of RICA against other popular ANNs. The test were performed using four different algorithms on two “real world” data sets and two sample data sets. The results for two of the data sets

are presented in the next section and Appendix A presents the other two. The algorithms included two fixed ANN architectures, an adaptive MLP, and RICA. The two fixed ANN algorithms were an RBF network and an MLP network. The hidden layer nodes of the RBF network were trained using a combination of a K -means algorithm and Mahalanobis distance clustering. The value for K was the same as the number of nodes that RICA determined was needed for each training set respectively. The K -means algorithm was used to find the centers of the hidden layer nodes and the Mahalanobis distance clustering was used to find the covariances. The output layer weights were calculated using the MSE pseudo-inverse approach described in Section 4.2. The fixed architecture MLP used a single layer of sigmoidal functions in the hidden layer nodes and sigmoids for the output layer nodes. The number of hidden layer nodes was the same as those found using MICA [68] for each training session and the MLP was trained using back-propagation with momentum. The adaptive MLP is called MICA [68]. Very simply, MICA is an adaptive MLP algorithm that is designed to grow the network architecture to optimize training set classification accuracy.

Four data sets were used to analyze the accuracy and computational costs of the four algorithms. The first set contained 3471 samples of 27 dimensional feature data extracted from multiple authors' hand-written characters 0 – 9. The second set contained samples of 2-dimensional three class data that was created using various distributions and centers, see Figure 4.3. The third set contained 5457 samples of 6 dimensional feature data extracted from infrared target imagery data. The fourth set was used to analyze the capabilities of the algorithms to separate two Gaussian sources. This data set and problem was first described by Kohonen [39] and was used by Lim [45] in his analysis of his Probabilistic Fuzzy ARTMAP Algorithm (PFAM) which was an extension to the Fuzzy ARTMAP algorithms(FAM). According to Lim this task is regarded as a benchmark study for statistical pattern recognition with neural networks. The task of separating two Gaussian sources which use this data set has three areas of difficulty:

1. Large overlapping of class distributions
2. A high degree of non-linearity in the decision boundaries
3. A high dimensionality of the input space.

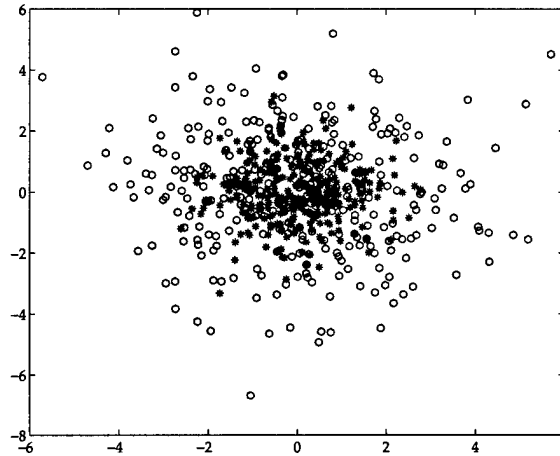


Figure 4.2 Plot of two dimensions of the benchmark data set. The benchmark data set is comprised of two multivariate Gaussians with centers $\mu_1 = \mu_2 = (0, 0, 0, 0, 0, 0, 0, 0)$ and variances of 1 and 4 along all dimensions, respectively

See Figure 4.2 for a plot of the 2-dimensional version of the benchmark data set.

The data were generated using a multivariate Gaussian distribution. There were two classes, $C1$ and $C2$, with mean vectors $\mu_1 = \mu_2 = (0, 0, 0, 0, 0, 0, 0, 0)$ and variances set equal to 1 and 4, respectively, for all dimensions. The training set consisted of 300 vectors and the test set consisted of 10,000 vectors with the above mentioned distributions. All of the algorithms previously described were tested using this data set and the results are tabulated in Table 4.2. In addition to the algorithms which were implemented and tested during this research, the results from Lim's work are also shown for comparison.

There were a total of 24 trials conducted using the Optical Character Recognition (OCR) data. Each consisted of randomly separating the data into a training set and a separate test set. The split was approximately 70 percent of the data for training and 30 percent for testing. There were a total of 30 trials using the second sample data set with a 70/30 split for training and testing. There were a total of 24 trials using the infrared imagery data using the same percentages of splitting the data. After each trial the accuracy and number of Floating Point Operations (FLOPs) were calculated. The average results are presented in the next section.

4.5 Results

Table 4.1 presents the average results of testing using the OCR data. Table 4.2 presents the average results of testing using the benchmark sample data. Included in Table 4.2 are also the results reported by Lim.

The test results were analyzed using a *t*-test [32] to determine if there was a statistically significant difference between the results of each algorithm and RICA. The *t*-test was accomplished in the following manner:

1. Create a vector for each algorithm whose elements contain the classification accuracies from each of the trial runs.
2. Subtract the results for RICA from the results for each of the other algorithms.
3. Generate the null hypothesis that the mean of the difference vector is zero (the results are identical between RICA and each of the other algorithms).
4. Perform the *t*-test using the null hypothesis. Decide, based upon a statistical significance of 95 percent, whether to reject the null hypothesis.

For example, if there were 10 trials and RICA had the following accuracies: [.98 .96 .97 .93 .95 .98 .96 .97 .93 .95] and MICA had [.97 .93 .98 .90 .91, .97 .93 .98 .90 .91] the difference vector is [.01 .03 -.01 .03 .04 .01 .03 -.01 .03 .04]. The results of the *t*-test on this example would be to reject the null hypothesis (i.e., the results of using the algorithms are not identical) at the significance level of .05 (a confidence of 95 percent).

In all cases, RICA was shown to outperform MICA, the other adaptive ANN, by a statistically significant amount and in most cases, RICA outperformed the fixed architecture RBF and MLP. As can be seen in Table 4.2, RICA also outperformed many other types of adaptive ANNs, which included:

1. The Probabilistic Neural Network (PNN)
2. The Minimum Error Neural Network (MNN)
3. The Fuzzy ARTMAP (FAM) Neural Network

Table 4.1 Results of 24 trials using an Optical Character Recognition feature data set.

Algorithm	Avg. Number of Nodes	Test Accuracy	Training FLOPs ($1e^8$)
MLP	69.5	0.9615	211
RBF	47.4	0.7103	8
MICA	69.5	0.9553	520
RICA	47.4	0.9655	136

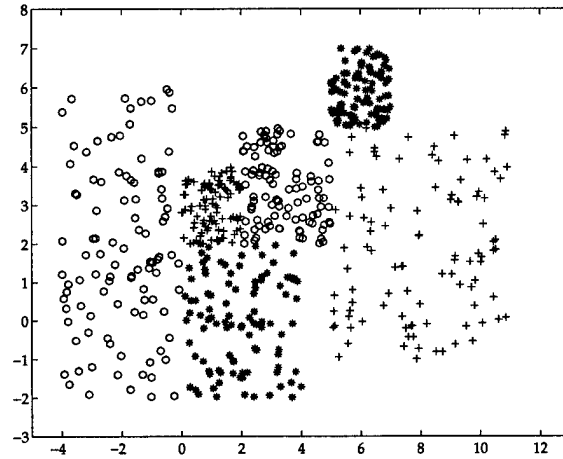


Figure 4.3 Plot of 3 class sample data.

4. The Probabilistic Fuzzy ARTMAP (PFAM) Neural Network

These algorithms are described by Lim in his paper about PFAM [45] and will not be discussed here. They are listed as a comparison to RICA. They were not actually implemented during this research. The results for these algorithms in Table 4.2 are reported from Lim's results.

In addition to outperforming the algorithms in classification accuracy, RICA also used less FLOPS than both MLP algorithms. The fixed architecture RBF algorithm used much fewer FLOPS than any of the other algorithms, but it needed to have the number of centers preselected.

4.6 Application to Automatic Target Cueing of Hyperspectral Image Data

The purpose of this section is to demonstrate the capabilities of combining the hyperspectral fusion algorithm, presented in Chapter III, with the Radial Basis Function Iterative

Table 4.2 Results of Using Bench Data with a known Bayes Optimal of a 91.0 percent accuracy rate.

Algorithm	Test Accuracy
Bayes Rate	91.0
MLP	79.5
RBF	73.0
PNN	74.3
FAM	75.0
MICA	82.0
PFAM	84.0
LVQ	86.6
MNN	87.0
RICA	90.0
BM2(Binary)	90.6

Constructive Algorithm (RICA), presented above, to perform Automatic Target Cueing of hyperspectral image data.

4.6.1 Motivation. As discussed in the Introduction, today's Air Force motto is "more with less." "More with less" means that the Air Force has to rely on modern technology to cover the losses of manpower experienced over the last decade. One area that technology can provide assistance is in the area of image analysis and target detection. The first part of the technology boost is in creating sensors that can provide vast amounts of both spatial and spectral information about a given target area. Hyperspectral sensors have been developed for both spatial and spectral information. The second part is in creating algorithms that can autonomously process and analyze the image data to provide either a human analyst or an Automated Target Recognition (ATR) algorithm with cues about possible targets in the image data. Automatic Target Cueing (ATC) allows the analyst or ATR algorithm to focus on the possible targets or regions-of-interest (ROIs) previously detected by the ATC algorithm. By focusing on ROIs detected by the ATC algorithm, instead of the whole image, the cost of using the much more labor intensive human analysis and the much more CPU intensive ATR algorithm can be reduced significantly. The results in this section demonstrate how fusion and ANNs can be used to perform ATC of hyperspectral image data.

4.6.2 *Experiments and Results.* Tests were conducted using 24 images from a hyperspectral image sensor. The test data was generated by using a human analyst to select coordinates in the imagery that represented both target and non-target areas. Targets could be any type of made-made object, while non-targets could be any non man-made object (i.e., grass, rocks, shrubs, trees, etc). The labeled coordinates were used to extract information from the images to create both a fused data set and a non-fused data set.

The non-fused data set was created by selecting sample values for several x,y coordinates from a given image using bands [20, 27, 30, and 58]. Each sample value represents the pixel values for that x,y coordinate and those four bands. Sampling along the four bands creates a labeled four dimensional sample vector for each x,y coordinate. These bands were chosen because they represent the center frequencies that have been shown to work well in detecting targets in multi-spectral data. The analysis showing their effectiveness was based on a signal-to-clutter ratio between targets and backgrounds. There were a total of 13,682 4-dimensional samples generated: 2862 target samples, and 10,820 non-target samples for the non-fused data set.

The fused data set was created using the same x,y coordinates as above, but this time the images were fused before the pixel values were selected. Instead of selecting samples from images representing bands [20, 27, 30, and 58], a set of bands around the neighborhood of bands [20, 27, 30, and 58] were first fused to form four new "bands". These new bands represent the information in:

- Bands [17, 18, 19, 20, 21, 22, 23]
- Bands [24, 25, 26, 27, 28]
- Bands [29, 30, 31, 32, 33]
- Bands [55, 56, 57, 58, 59, 60, 61].

The four fused images represent the information contained in the 24 images that were fused. This is very useful to image analysts. Instead of analyzing 24 images they only have to analyze four. Target recognition algorithms also benefit from the fusion. If the original 24 images were used, the data vectors would be 24 dimensional. Fusion reduces

Table 4.3 Results of 30 trials using the Non-fused and Fused data sets.

Type	Class 1 (Targets) Avg. Accuracy	Class 2 (Non-targets) Avg. Accuracy
Non-fused	46.3	66.8
Fused	65.1	73.7

the number to 4 dimensions while maintaining the information in the 24 images. This reduces both the number of CPU cycles required (less calculations are needed) and reduces the chance of the “curse of dimensionality” [7,29] affecting the ability of the algorithm to generalize. Less calculations are required because the clustering algorithm and the transformation at the hidden layer nodes will be operating on 4-dimensional data instead of 24-dimensional data. The “curse of dimensionality” can occur in two ways. The first is that as the number of dimensions increases the complexity or number of calculations increases, this can possibly exceed the computer’s computing or memory capabilities [7]. The second is that density estimation becomes more difficult and requires many more samples as the dimension increases [29]. Using fusion to reduce the dimension of the data from 24 to 4 reduces the risk of the “curse of dimensionality”.

After the data sets were created, 30 separate trials were performed using a 50/50 split of the data. Table 4.3 presents the average results of the 30 trials using both the non-fused and fused data. As with the other tests conducted in this research, the paired t-test was used to determine if the improvement was statistically significant [32]. The results show that target detection using RICA and fusion together provide a statistically significant improvement over target detection with RICA alone. Fusion provided a gain of almost 20 percentage points on average in target recognition accuracy.

The main reason for the previous tests was to analyze RICA’s ability to perform target/non-target recognition of fused and non-fused sample hyperspectral data. The tests clearly demonstrated that recognition accuracy is significantly improved using image fusion. The next set of tests will demonstrate how physiologically motivated image fusion can be used in conjunction with RICA to perform ATC of hyperspectral data.

Because ATC is performed, the results of these tests will be analyzed in terms of detection accuracy versus false alarm rate, instead of classification accuracy. The difference between detection and classification accuracy is that classification accuracy is measured at the pixel level and detection accuracy is measured at the object level. Classification accuracy is a ratio of the number of correctly classified samples to the number of samples for that class. Detection accuracy is a ratio of the number of times that an object has been detected to the number of objects available to detect. For example, given an image that contained one target with 45 pixels on the target, if 32 of those pixels are correctly classified the classification accuracy is 71.1 percent. The detection accuracy is 100 percent. In other words, there was one target and it was recognized. It does not matter how many of the pixels are recognized. What matters is that at least one of the pixels on the target were correctly identified. If there were 5 targets and 3 of them were identified, then the detection accuracy would be 60 percent. As mentioned, when ATC algorithms are analyzed, the results are typically reported in terms of a detection accuracy versus false alarm rate.

The false alarm rate is the average number of ROIs that are falsely declared as targets per image. For example, a 98 percent detection accuracy with a false alarm rate of 3 ROIs per image, means that 98 percent of the time the target is correctly detected in an image, but on average there are 3 ROIs per image that are falsely declared as targets. In this analysis, a ROI is defined as a 30×30 box centered at a pixel that has been declared as a target by RICA. If the box contains other pixels that were declared as a target by RICA, then it is still only one ROI. The reason a 30×30 box was chosen is because the images were collected at a range such that the average number of pixels on target is roughly 30×30 .

This type of measurement is very important to an analyst when analyzing an ATC algorithm. It does not help the analyst if the detection accuracy is high when there are also a lot of false alarms. The goal of ATC is to reduce the burden of the analyst. Therefore, the goal is to reduce false alarms, while maintaining or increasing accuracy. The following tests will demonstrate how RICA and image fusion work together to provide excellent target detection while significantly reducing false alarms.

The tests are conducted using the same 23 hyperspectral images used in the previous tests, the same fused and non-fused images that were extracted from them, and the same fused and non-fused data sets. The difference between these tests and the previous tests is that once RICA has been trained using the fused and non-fused data sets, it will be tested on the whole image instead of only a small subsample of the image. The tests above were conducted using only a fraction of the samples available in these images. On average only three percent of the available pixels of each image were sampled, labeled, and used for training/testing. This was sufficient to measure classification accuracy. To analyze detection accuracy versus false alarm rate, the whole image will need to be tested.

In previous tests, classification (i.e., target/non-target) was determined by whatever node had the greatest output. If node one was greater, the input sample was classified as a target. If node two was greater, the input sample was classified as non-target. In an effort to reduce false alarms, the outputs of RICA are thresholded and both output nodes have to strongly agree that a sample is a target before it will be classified as a target. What this means and how the tests were conducted is described next.

After training two RICAs (one on the non-fused and one on the fused data set), the following steps are performed for each hyperspectral image, also see Algorithms 3 and 4 for a pseudo code description of the process.

1. Extract the 4-dimensional sample vectors from every x,y coordinate from the non-fused bands [20, 27, 30, and 58] of a given image.
2. Record the outputs of RICA for each of the sample vectors for that image, using the RICA that was trained on the non-fused data.
3. Calculate the mean and standard deviation of the total outputs of each node.
4. Classify any vector as a target that has an output from node one that is two standard deviations above the mean of node one and at the same time has an output from node two that is two standard deviations below the mean of node two. This thresholding allows only those samples that RICA strongly declares as a target and as not a non-target to be identified as a target.

Algorithm 3 Target Detection of Non-Fused Hyperspectral Images

$I_n \triangleq X \times Y \times B$ Hyperspectral Image Cube for image n
 $I_{n,20,27,30,58} \triangleq$ Hyperspectral Image Cube for image n {contains only bands [20,27,30,58]}
 $A_n \triangleq X \times Y$ Image representing the ATC of I_n
 $R \triangleq$ Matrix containing outputs of RICA for every test sample (one result per row)
 $Rout(t) \triangleq$ output vector representing the output of RICA for test sample t
 $I_{n,20,27,30,58}(x, y, :) \triangleq$ vector represents the pixel values at location $x, y, [20, 27, 30, 58]$ {of a hyperspectral image}
 $X \leftarrow$ Number of Rows of the Hyperspectral Image for each band
 $Y \leftarrow$ Number of Columns of the Hyperspectral Image for each band
 $B \leftarrow$ Number of Bands in the Hyperspectral Image cube
 $N \leftarrow$ Number of Hyperspectral Image Cubes
for $n \leftarrow 1 : N$ **do**
 $i \leftarrow 0$
 for $x \leftarrow 1 : X$ **do**
 for $y \leftarrow 1 : Y$ **do**
 $i \leftarrow i + 1$
 $R(i, :) \leftarrow Rout(I_{n,20,27,30,58}(x, y, :))$
 end for
 end for
 $M_R \leftarrow mean(R)$ {This is the mean calculated for each column (i.e., each class)}
 $S_R \leftarrow std(R)$ {This is the standard deviation calculated for each column (i.e., each class)}
 $A_n \leftarrow 0_{X,Y}$ {Initialized to all zeros}
 $i \leftarrow 0$
 for $x \leftarrow 1 : X$ **do**
 for $y \leftarrow 1 : Y$ **do**
 $i \leftarrow i + 1$
 if $[R(i, 1) > M_R(1) + 2 * S_R(1)] \& [R(i, 2) < M_R(2) + 2 * S_R(2)]$ **then**
 $A_N(x, y) \leftarrow 1$ {Is Target}
 else
 $A_N(x, y) \leftarrow 0$ {Is Not Target}
 end if
 end for
 end for
end for

Algorithm 4 Target Detection of Fused Hyperspectral Images

$I_n \triangleq X \times Y \times B$ Hyperspectral Image Cube for image n
 $I_{n,20,27,30,58} \triangleq$ Hyperspectral Image Cube for image n {contains the four images representing the fused bands centered at bands [20,27,30,58]}
 $A_n \triangleq X \times Y$ Image representing the ATC of I_n
 $R \triangleq$ Matrix containing outputs of RICA for every test sample (one result per row)
 $Rout(t) \triangleq$ output vector representing the output of RICA for test sample t
 $I_{n,20,27,30,58}(x, y, :) \triangleq$ vector represents the pixel values at location $x, y, [20, 27, 30, 58]$ {of the fused hyperspectral image}
 $X \leftarrow$ Number of Rows of the Hyperspectral Image for each band
 $Y \leftarrow$ Number of Columns of the Hyperspectral Image for each band
 $B \leftarrow$ Number of Bands in the Hyperspectral Image cube
 $N \leftarrow$ Number of Hyperspectral Image Cubes
for $n \leftarrow 1 : N$ **do**
 $i \leftarrow 0$
 for $x \leftarrow 1 : X$ **do**
 for $y \leftarrow 1 : Y$ **do**
 $i \leftarrow i + 1$
 $R(i, :) \leftarrow Rout(I_{n,20,27,30,58}(x, y, :))$
 end for
 end for
 $M_R \leftarrow mean(R)$ {This is the mean calculated for each column (i.e., each class)}
 $S_R \leftarrow std(R)$ {This is the standard deviation calculated for each column (i.e., each class)}
 $A_n \leftarrow 0_{X,Y}$ {Initialized to all zeros}
 $i \leftarrow 0$
 for $x \leftarrow 1 : X$ **do**
 for $y \leftarrow 1 : Y$ **do**
 $i \leftarrow i + 1$
 if $[R(i, 1) > M_R(1) + 2 * S_R(1)] \ \& \ [R(i, 2) < M_R(2) + 2 * S_R(2)]$ **then**
 $A_N(x, y) \leftarrow 1$ {Is Target}
 else
 $A_N(x, y) \leftarrow 0$ {Is Not Target}
 end if
 end for
 end for
end for

5. Create an image representing the classification results.
6. Extract the 4-Dimensional sample vectors from every x,y coordinate from the fused bands of a given image.
7. Record the outputs of RICA for each of the sample vectors for that image, using the RICA that was trained on the fused data.
8. Calculate the mean and standard deviation of the total outputs of each node.
9. Classify any vector as a target that has an output from node one that is two standard deviations above the mean of node one and at the same time has an output from node two that is two standard deviations below the mean of node two. This thresholding allows only those samples that RICA strongly declares as a target and as not a non-target to be identified as a target.
10. Create an image representing the classification results.

The analysis was conducted image by image and detection rates are presented in Table 4.4. The most significant results of performing ATC using RICA and physiologically motivated image fusion is in the reduction of false alarms. When comparing the ATC results of the non-fused images with that of the fused images, it is easy to see that the number of false ROIs in the fused images are much less than the Non-Fused image results (white spots indicate regions that have been declared as targets). It is also important to point out that the reduction in false alarms was accomplished without sacrificing detection capability.

The results of Table 4.4 show that on average target detection is improved with fusion. It shows that the average detection rate with fusion is 68.2 percent which is almost 5 percentage points better than the non-fused result of 63.6 percent. In addition to an average improvement in detection rates, a statistically significant reduction in false alarms is achieved. The false alarm rate is reduced from 5.84 ROIs per image to 3.42. The reduction in false alarms means that an image analyst has to spend less time looking in areas that do not contain targets.

Again, these tests were intended as an example of the ATC capabilities of the combination of RICA and the physiologically motivated fusion algorithm developed for this research.

Table 4.4 ATC Results Using 21 Hyperspectral Images.

Type	Detection Rate Avg.	False Alarms Avg. Per Image
Non-fused	63.6	5.84
Fused	68.2	3.42

Analyzing the results show that ATC significantly reduces the number of areas of an image that have to be analyzed. The overall results clearly demonstrate that the combination of RICA and physiologically motivated image fusion provides an excellent means of performing ATC of hyperspectral image data.

4.7 Conclusions

The Radial Basis Function Iterative Construction Algorithm (RICA) which was designed and implemented in this research has been demonstrated to outperform many types of ANNs including RBF, MLP, LVQ, MNN, PNN, FAM, PFAM, and MICA. These algorithms included both fixed and adaptive types of architectures. It outperformed many of these algorithms in both classification accuracy and in FLOPs. In the limited number of cases where the fixed architecture algorithms outperformed RICA, it is important to note that the number of nodes needed had to be provided by a human.

This chapter also provided a mathematical proof showing that RICA performs a minimum MSE approximation to a Bayes Optimal Classifier. The results of Table 4.2 support this proof with experimental evidenced. The Bayes Optimal Classification accuracy is 91.0 percent and RICA had a 90.0 percent accuracy. The only algorithm to outperform RICA on the benchmark data set was the Boltzmann machine with binary-coded input vectors (BM2). It had an accuracy of 90.6 percent. The BM2 process required user input to determine the number of subranges to binarize the input data and the number of nodes were also preselected by a human.

As mentioned previously, RICA also outperformed most of the algorithms in FLOPS. The only algorithm RICA did not outperform in FLOPS was the fixed architecture RBF. Again, note that the fixed architecture algorithm had to have the number of nodes prese-

lected, RICA does not. The reduction in FLOPS is due to the local nature of the radial basis functions. For a standard MLP, back-propagation has to perform an iterative update based upon the error associated with the hidden layer and then with the output layer. The two layers can not be optimized independently because of the global nature of Sigmoid basis functions. RICA does not suffer from this limitation. The local nature of Radial Basis Functions allows RICA to optimize the hidden layer nodes and then the output layer nodes [9]. Another reason RICA outperforms the standard MLP using back-propagation is that the sigmoid unit has a very slow learning rate once the sigmoid is into the saturation region (i.e., the slope of the curve is effectively zero at the saturation regions). The other problem with the global nature of sigmoids is that minimization using back-propagation does not guarantee a global minimization. There are often problems with falling into a local minima. RICA does not suffer from this effect. Minimizing RICA on a local scale (i.e., at each basis function and then at the output layer) provides a global minimization. Again, this is evidenced by both the Bayes proof and the results of Table 4.2.

The reason RICA outperforms MICA in flops is that RICA's method of optimizing the hidden layer basis functions uses a much more efficient algorithm than MICA does. MICA requires a search through all data points to continually find the next data pair that has the smallest Euclidean distance. RICA evaluates each class separately and once a set of points has been partitioned and associated with a basis function, they are eliminated from the search. Therefore, RICA breaks the problem into smaller and smaller chunks, while MICA and back-propagation continually iterate through the whole data set.

It is important to note that the current version of RICA uses the univariate test for each dimension and averages across the results to determine the goodness-of-fit. This works well when the features are independent. If they are not, there may be some loss of power [18] with this method. Future work should implement a method by Malkovich and Afifi [48] that is a true multivariate Gaussian goodness-of-fit measure.

The results of Section 4.6 provided evidence that the physiologically motivated fusion algorithm and RICA can be used in combination to perform ATC of hyperspectral image data. The results presented in Table 4.3 show that automatic target cueing using RICA and

Fusion together provide a significant improvement over target detection with RICA alone. In this example, image Fusion provided a gain of almost 20 percentage points in target detection accuracy. Additionally, the results in Table 4.4 show that the combination of RICA and physiologically motivated image fusion provides an excellent means of performing ATC of hyperspectral image data.

In summary, RICA has been shown, both mathematically and empirically, to be a very powerful self-architecting ANN, capable of statistically outperforming some of the most popular ANNs on several types of data sets. It has also been demonstrated that RICA can be used in combination with the physiologically motivated fusion algorithm to perform ATC of hyperspectral image data.

V. Conclusions and Contributions

5.1 Conclusions

Modern imaging sensors produce vast amounts of information about a given scene, overwhelming both human analysts and automatic target recognition systems. Autonomous systems are required that can fuse and then classify if we are going to exploit all of this data. The results presented in Section 4.6 provide a clear example that the combination of physiologically motivated fusion and the self-architecting capabilities of RICA provide a very effective means of autonomously processing the vast amounts of hyperspectral data.

In addition to having the ability to fuse and then classify hyperspectral data, the physiologically motivated fusion algorithm and RICA have also been proven to outperform current algorithms in their respective classes. For example, it was shown in Chapter III that the novel fusion algorithm designed and implemented for this research outperformed all of the image fusion algorithms in terms of both signal-to-noise ratios and image aesthetics. This was a clear demonstration that a combination of wavelet-based multiresolution analysis weighted by the human visual system's contrast sensitivity function is an effective means to fuse images while suppressing noise and maintaining image detail. The results in Table 4.3, which show an increase in target classification accuracy of 19 points and a decrease in the false alarm rate by 7 points after fusion, provide another example of how this fusion algorithm suppresses noise and increases image detail. What Table 4.3 does not show is the unique way in which RICA performs classification. Namely, RICA autonomously adapts its architecture to optimize classification accuracy for a given data set. The fact that RICA is very good at this is demonstrated in Chapter IV, where RICA is shown to outperform many ANN architectures, including both fixed and adaptive. It is also important to note that not only did Chapter IV provide empirical evidence that RICA is an excellent self-architecting ANN, it also presented a proof showing that RICA performs a minimum mean squared-error approximate to a Bayes optimal classifier and that the outputs of RICA can be interpreted as *a posteriori* probabilities. Therefore, a level of confidence may be associated

with the classification of each sample which could prove very useful in accepting/rejecting a classification or in using RICA in a hierarchical network.

5.2 Contributions

A comprehensive method to autonomously fuse and classify hyperspectral sensor data was introduced. The capabilities of this method were demonstrated using real world hyperspectral image data. These demonstrations showed that a combination of physiologically motivated image fusion and a self-architecting ANN are very effective in processing the vast amounts of image data produced by hyperspectral sensors.

During the process of developing the fusion and classification model, a novel physiologically motivated fusion algorithm was developed and implemented. The physiologically motivated fusion algorithm demonstrated the usefulness of combining a wavelet-based multiresolution analysis with the contrast sensitivity function of the human visual system. This fusion algorithm was shown to outperform other fusion algorithms in both signal-to-noise ratios and image aesthetics. It was shown that contrast sensitivity based fusion, in conjunction with wavelet-based decomposition and reconstruction, results in better SNRs over a wide range of noise types, levels, and backgrounds and that the results are independent of the type (correlated vs uncorrelated) and strength of the added noise. It was also shown that this new fusion method statistically outperformed other fusion algorithms by an average of 6.8db in SNRs. In addition, the fusion algorithm was proven to be useful in fusing hyperspectral imagery for both human analysis and autonomous target detection algorithms. It was shown that an increase of almost 20 percentage points on average in detection accuracy was achieved using this fusion algorithm.

In addition to developing a very effective fusion algorithm, a unique self-architecting ANN RICA was designed and implemented. RICA was designed to autonomously determine its architecture to optimize classification accuracy. Its capabilities were demonstrated using a wide range of both contrived and measured data sets. Using these data sets, RICA was shown to statistically outperform many other neural and statistical classifiers. It is important to note that in the cases where another ANN achieved a better classification accuracy

than RICA, the ANN's architecture had to be preselected. The goal of this research was an autonomous system. RICA was also proven to be useful in classifying fused hyperspectral imagery data. As mentioned previously, the combination of fusion and RICA resulted in an increase in target detection of nearly 20 points. This increase in detection was also accompanied by a decrease in the false alarm rate of 7 points. Not only was the superior performance of the self-architecting ANN RICA empirically demonstrated, it was mathematically proven to be a minimum mean squared-error approximation to a Bayes optimal classifier. Also, it was mathematically proven to have outputs that can be interpreted as *a posteriori* probabilities. This is very useful in assigning confidences to the classifications and in using RICA's outputs in hierarchical classifiers.

In summary, this dissertation has presented a novel Bayes optimal self-architecting ANN that is very capable of classifying the outputs of a superior physiologically motivated fusion algorithm.

5.3 Recommendations for Future Work

The current version of RICA uses a univariate test for each dimension and averages across the results to determine the goodness-of-fit. This works well when the features are independent. If they are not, there may be some loss of power [18] with this method. Future work should implement a method by Malkovich and Afifi [48] that is a true multivariate Gaussian goodness-of-fit measure. Another suggestion for future work is to combine both Gaussian and Sigmoidal basis functions in the hidden layer.

Appendix A. Additional Test Results

This appendix presents the results of testing on two addiotnal data sets using RICA. Table A.1 presents the average results of testing using the sample data. Table A.2 presents the average results of testing using the infrared imagery data.

Table A.1 Results of 30 trials using 3 class sample test data.

Algorithm	Avg. Number of Nodes	Test Accuracy	Training FLOPs ($\times 10^7$)
MLP	14.5	0.9561	10
RBF	56.9	0.9776	4
MICA	14.5	0.9463	100
RICA	56.9	0.9663	19

Table A.2 Results of 24 trials using Infrared Target imagery.

Algorithm	Avg. Number of Nodes	Test Accuracy	Training FLOPs ($\times 10^8$)
MLP	47.6	0.6645	62
RBF	185.6	0.6563	39
MICA	47.6	0.5861	211
RICA	134.5	0.6198	118

Bibliography

1. Akerman, Alexander. "Pyramidal Techniques for Multisensor Fusion," *Proceedings of SPIE - The International Society for Optical Engineering*, 1828:124-131 (1993).
2. Alexander Toet, L. J. van Ruyven, J. M. Valetton. "Merging Thermal and Visual Images by a Contrast Pyramid," *Optical Engineering*, 28(7):789-792 (1989).
3. Anderson, Nonboe, et al. "Adaptive Regularization of Neural Classifiers." *Neural Networks for Signal Processing VII.*, edited by J. Principe, et al. 24-33. 1997.
4. Baum, E. B. "On the Capabilities of Multilayer Perceptrons," *Journal of Complexity*, 4:193-215 (1988).
5. Baum, E. B. "What Size Net Gives Valid Generalization," *Neural Computation*, 1:151-160 (1989).
6. Baum, E. B. and D. Haussler. "What size net gives valid generalization," *Neural Computation*, 1:151-160 (1989).
7. Bellman, R. *Adaptive Control Processes: A Guided Tour.*, Princeton University Press, 1961.
8. Berthold, Michael. "A probabilistic Extension for the DDA Algorithm." *The 1996 IEEE International Conference on neural networks ICNN*. 341-346. June 1996.
9. Bishop, Christopher M. *Neural Networks for Pattern Recognition*. Clarendon Press, 1996.
10. Burt, Peter J. and Edward H. Adelson. "The Laplacian Pyramid as a Compact Image Code," *IEEE Transactions on Communications*, 31(4):532-540 (April 1983).
11. Burt, Peter J. and Edward H. Adelson. "Merging Images Through Pattern Decomposition," *SPIE, Applications of digital image processing*, 575(3):173-181 (1985).
12. Burt, Peter J. and Raymond J. Kolczynski. "Enhanced Image Capture Through Fusion," *1993 IEEE 4th International Conference on Computer Vision*, 4:173-182 (1993).
13. Carpenter, Gail, et al. "ARTMAP: Supervised Real-Time Learning and Classification of Non-stationary Data by a Self-Organizing Neural Network," *Neural Networks*, 4:565-588 (1991).
14. Chakravarthy, Srinivasa V. and Joydeep Ghosh. "Function Emulation Using Radial Basis Function Networks," *Neural Networks*, 10(3):459-478 (1997).
15. Choi, Chong-Ho and Jin Young Choi. "Constructive Neural networks with Piecewise Interpolation Capabilities for Function Approximation," *IEEE Transactions on Neural Networks*, 5(6):936-944 (November 1994).
16. Clippingdale, Simon and Roland Wilson. "Self-Similar Neural Networks Based on a Kohonen Learning Rule," *Neural Networks*, 9(5):747-763 (1996).

17. Cybenko, George. "Approximation by Superposition of Sigmoidal Functions," *Mathematics of Control Signals, and Systems*, 2:303-314 (1989).
18. D'Agostino, Ralph and Michael A. Stephens, editors. *Goodness-Of-Fit Techniques*, chapter 5 and 9, 195-230, 367-413. Marcel Dekker, Inc., 1986.
19. Daubechies, Ingrid. *Ten lectures on wavelets*. Philadelphia: Society for Industrial and Applied Mathematics, 1992.
20. Duda, Richard O. and Peter E. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, 1973.
21. Fielding, Kenneth Henry. *Spatio-Temporal Pattern Recognition Using Hidden Markov Models*. PhD dissertation, Air Force Institute of Technology, 1994.
22. Fletcher, J. and Z. Obradovic. "Parallel and Distributed Systems for Constructive Neural Network learning." *Proceedings of the 2nd International Symposium on High Performance Distributed Computing*. 174-186. July 1993.
23. Fletcher, Justin and Zoran Obradovic. "Constructively learning a near-Minimal neural network Architecture." *1994 IEEE International Conference on Neural Networks. IEEE World Congress on Computational Intelligence* 1. 204-208. 1994.
24. Fletcher, Justin and Zoran Obradovic. "A Discrete Approach to Constructive neural Network Learning," *Neural, Parallel and Scientific Computations*, 3:307-320 (1995).
25. Foley, Donald H. "Considerations of Sample and Feature Size," *IEEE Transactions on Information Theory*, IT-18(5):618-626 (September 1972).
26. Fort, Jean-Claude and Gilles Pages. "About the Kohonen Algorithm: Strong or Weak Self-Organization?," *Neural Networks*, 9(5):773-785 (1996).
27. Fritzke, Bernd. "Growing Cell Structures - A self-Organizing Network for Unsupervised and Supervised Learning," *Neural networks*, 7(9):1441-1460 (1994).
28. Geman, S., et al. "Neural Networks and the Bias/Variance Dilemma," *Neural Computation*, 4:1-58 (1992).
29. Hanka, Rudolf and Thomas Harte. "Curse of Dimensionality: Classifying Large Multi-Dimensional Images with Neural Networks." *Proceedings of the IEEE European Workshop on Computer Intensive Methods in Control and Signal Processing*. 1996.
30. Hartman, Eric J. and James D. Keeler. "Layered Neural Networks with Gaussian Hidden Units as Universal Approximations," *Neural Computation*, 2(2):210-215 (Summer 1990).
31. Hinton, Geoffrey E. *How Neural Networks Learn from Experience*, chapter 10, 113-124. W.H. Freeman and Company, 1993.
32. Hogg, Robert V. and Allen T. Craig. *Introduction To Mathematical Statistics*. Prentice-Hall, Inc., 1995.

33. Jain, Anil K. and Jianchang Mao. "Neural Networks and Pattern Recognition." *Computational Intelligence Imitating Life* edited by Jacek M. Zurada, et al., chapter 4, 194-212, Institute of Electrical and Electronics Engineers IEEE Press, 1994/1994.
34. Jr., Mark W. Cannon. "A Study of Stimulus Range Effects in Free Modulus Magnitude Estimation of Contrast," *Vision Research*, 24(9):1049-1055 (1984).
35. Kangas, Jari, et al. "Variants of Self-Organizing Maps," *IEEE Transactions on Neural Networks*, 1(1):93-99 (1990).
36. Karayiannis, Nicolaos B. and Glenn Weiqun Mi. "Growing Radial Basis Neural Networks: Merging Supervised and Unsupervised Learning with Network Growth Techniques," *IEEE Transactions on Neural Networks*, 8(6):1492-1506 (November 1997).
37. Kobayashi, Kunikazu, et al. "Fundamental Consideration on Self-Formation of Recognition Cells," *Neural Networks*, 7(8):1241-1252 (1994).
38. Kohonen, Teuvo. *Self-Organization and Associative Memory*. Springer-Verlag, 1989.
39. Kohonen, Teuvo, et al. "Statistical Pattern Recognition with Neural Networks: Benchmarking Studies." *Proceedings of the IEEE International Conference on Neural Networks I1*. 61-68. 1988.
40. Kohonen, Teuvo. "Improved Versions of Learning Vector Quantization." *1990 IEEE INNS International Joint Conference on Neural Networks1*. 545-550. 1990.
41. Kohonen, Teuvo. *Statistical Pattern Recognition Revisited*. New York: Elsevier Science, 1990.
42. Krzyzak, A., et al. "Non-parametric Classification Using Radial Basis Function nets and Empirical Risk Minimization." *Proceedings of the 12th IAPR International Conference on Pattern Recognition2*. October 1994.
43. LeCun, Y., et al. "Back-propagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, 1(4):541-551 (Winter 1989).
44. LeCun, Y, et al. "Optimal Brain Damage." *Advances in Neural Information Processing Systems2*, edited by D. Touretzky. 598-605. 1990.
45. Lim, Chee and Robert Harrison. "An Incremental Adaptive Network for On-Line Supervised Learning and Probability Estimation," *Neural Networks*, 10(5):925-939 (1997).
46. Linde, Y., et al. "An Algorithm for Vector Quantizer Design," *IEEE Transactions on Communications*, COM-28:84-94 (January 1980).
47. Lowe, David. "Extracting Structure from Wake EEG Using Neural Networks." *Proceedings of the SPIE - The International Society for Optical Engineering3077*. 17-26. SPIE-Int. Soc. Opt. Eng, 1997.
48. Malkovich, J and A. Afifi. "On Tests for Multivariate Normality," *Journal of the American Statistical Association*, 68(341):176-179 (March 1973).

49. Mallat, Stephane G. "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674-693 (July 1989).
50. Mannos, James L. and David J. Sakrison. "The Effects of a Visual Fidelity Criterion on the Encoding of Images," *IEEE Transactions on Information Theory*, IT-20(4):525-536 (July 1974).
51. Mao, Jianchang and Anil Jain. "A Self-Organizing Network for Hyperellipsoidal Clustering (HEC)," *IEEE Transactions on Neural Networks*, 7(1):16-29 (January 1996).
52. Mascioli, F.M. Frattale, et al. "Comparisons of Constructive Algorithms for Neural Networks." *ICANN '94 Proceedings of the International Conference on Artificial neural Networks*. 731-734. 1994.
53. Moody, John and Christian Darken. "Fast Learning in Networks of Locally-Tuned Processing Units," *Neural Computation*, 1 (1989).
54. Moore, Barbara. "ART 1 and Pattern Clustering." *Proceedings Connectionist Summer School*. 174-185. 1988.
55. NagaBhushana, T. and H. Chandrasekharaiah. "Fault Diagnosis of Electrical Power Systems Using Incremental Radial Basis Function Nets." *EMPD'95 1995 International Conference on Energy Management and Power Delivery*. 560-564. November 1995.
56. NG, C. K., et al. "Constructive Learning Algorithm of Multilayer Neural network Using Linear Least Square Technique." *Proceedings of the International Conference on Neural Information Processing ICONIP 96*. 1996.
57. Pal, Nikhil, et al. "Sequential Competitive Learning and the Fuzzy c-means Clustering Algorithms," *Neural Networks*, 9(5):787-796 (1996).
58. Papoulis, Athanasios. *Probability, Random Variables, and Stochastic Processes*. New York, NY: McGraw-Hill, 1991.
59. Park, J. and I. Sandberg. "Universal Approximation Using Radial-Basis-Function Networks," *Neural Computation*, 3(2):246-257 (Summer 1991).
60. Parzen, E. "On Estimation of a Probability Density Function and Mode," *Annals of Mathematical Statistics*, 33:1065-1076 (1962).
61. Peli, Eli. "Contrast in Complex Images," *Optical Society of America*, 7(10):2032-2040 (October 1990).
62. Peli, Eli, et al. "Contrast Sensitivity Functions for Analysis and Simulation of Visual Perception," *Optical Society of America*, 3:126-129 (February 1990). Non-invasive Assessment of the Visual System 1990 Technical Digest Series.
63. "Playboy Magazine," nov 1972.
64. Poggio, Tomaso and Federico Girosi. "Networks For Approximation and Learning," *Proceedings of the IEEE*, 78(9):1481-1497 (September 1990).

65. Polakowski, W., et al. "Computer-aided breast cancer detection and diagnosis of masses using difference of Gaussians and derivative-based feature saliency," *IEEE Transactions on Medical Imaging*, 16(6):811-819 (1997).
66. Powell, J. M. D. "Restart Procedures for the Conjugate Gradient Method," *Mathematical Journal*, 12:241-254 (1977).
67. Priddy, Kevin L. *Feature Extraction and Classification of FLIR Imagery Using Relative Locations of Non-Homogeneous Regions with Feed-forward Neural Networks*. PhD dissertation, Air Force Institute of Technology, WPAFB, OH, April 1992.
68. Rathbun, Thomas, et al. "MLP iterative construction algorithm," *Neurocomputing*, 17:195-216 (1997).
69. Reed, R. "Pruning algorithms - a survey," *Neural Networks*, 4(5):740-747 (Sept 1993).
70. Rementeria, Santiago and Xabier Olabe. "On Simultaneous Weight and Architecture Learning." *Biological and Artificial Computation: From Neuroscience to Technology IWANN '97*. 501-509. Springer, June 1997.
71. Richard, Michael and Richard Lippmann. "Neural Network Classifiers Estimate the Bayesian *a Posteriori* Probabilities," *Neural Computation*, 3:461-483 (1991).
72. Rinker, Jack N. *Hyperspectral Imagery - What Is It? - What Can It Do?*. DTIC AD-A231 164, U.S. Army Engineer Topographic Laboratories, 1990.
73. Rogers, S., et al. "Neural Networks for Automatic Target Recognition," *Neural Networks*, 8(7,8):1153-1184 (1995).
74. Rogers, S., et al. "Artificial Neural Networks for Early Detection and Diagnosis of Cancer," *Cancer Letters*, 77(2,3):79-83 (March 1994).
75. Rogers, Steven K., et al. *An Introduction to Biological and Artificial Neural Networks*. Air Force Institute of Technology, 1990.
76. Rosenblatt. *Principles of Neurodynamics*. New York, NY: Spartan Books, 1959.
77. Rosenfeld, A., editor. *Multiresolution Image Processing and Analysis*. New York: Springer-Verlag, 1984.
78. Roy, Asim, et al. "An Algorithm to Generate Radial Basis Functions (RBF)-Like Nets for Classification Problems," *Neural Networks*, 8(2):179-201 (1995).
79. Rubin, Gary S. and Ronald A. Schuchard. "Does Contrast Sensitivity Predict Face Recognition Performance in Low-Vision Observers," *Optical Society of America*, 3:130-133 (February 1990). Non-invasive Assessment of the Visual System 1990 Technical Digest Series.
80. Ruck, Dennis W. and Steven Rogers. "Feature Selection Using a Multilayer Perceptron," *Journal of Neural Network Computing*, 40-48 (Fall 1990).
81. Ruck, Dennis W., et al. "The Multilayer Perceptron as an Approximation to a Bayes Optimal Discriminant Function," *IEEE Transactions on Neural Networks*, 1(2):296-298 (December 1990).

82. Ruck, Dennis W., et al. "Multisensor Fusion Classification with a Multilayer Perceptron." *Proceedings of IEEE/INNS International Joint Conference on Neural Networks*. 863-868. June 1990.
83. Rumelhart, David E., et al. *Learning Internal Representations by Error Propagation*, September 1985. Technical Report, University of California, San Diego: Institute for Cognitive Science, September 1985.
84. Sanders, Mark S. and Ernest J. McCormick. *Human Factors in Engineering and Design* (7th Edition). New York: McGraw-Hill, Inc, 1993.
85. Schalkoff, Robert J. *Pattern Recognition Statistical, Structural, and Neural Approaches*. John Wiley and Sons, Inc., 1992.
86. Shapiro, S. and R. Francia. "Approximate Analysis of Variance test for Normality(complete samples)," *Journal of the American Statistical Association*, 67:215-216 (1972).
87. Shapiro, S. and M. Wilk. "An Analysis of Variance Test for Normality," *Biometrika*, 52:591-611 (1965).
88. Smagt, Patrick P. Van Der. "Minimization Methods for Training Feed-forward Neural Neural Networks," *Neural Networks*, 7(1):1-11 (1994).
89. Stager, P. and D. Hameluck. "Contrast Sensitivity and and Visual Detection in Search and Rescue." *Defense and Civil Institute of Environmental Medicine*. June 1986.
90. Steppe, Jean. *Feature and Model Selection in Feed-forward Neural Networks*. PhD dissertation, Air Force Institute of Technology (AFIT), 1994.
91. Steppe, Jean, et al. "Integrated Feature and Architecture Selection," *IEEE Transactions on Neural Networks*, 7(4):1007-1014 (July 1996).
92. Tarassenko, L. and S. Roberts. "Supervised and Unsupervised Learning in Radial Basis Function Classifiers," *IEE Proceedings-Vision, Image and Signal Processing*, 141(4):210-216 (August 1994).
93. Toet, Alexander. "Hierarchical Image Fusion," *Machine Vision and Applications*, 1-11 (1990).
94. Toet, Alexander. "Multiscale Contrast Enhancement with Applications to Image Fusion," *Optical Engineering*, 31(5):1026-1031 (May 1992).
95. Widrow, B. "Adaline and Madaline-1963 (neural networks)." *IEEE First International Conference on Neural Networks* 1. 145-57. 1987.
96. Wilson, Terry, et al. "Perceptual Based Image Fusion For Hyperspectral Data," *IEEE Transactions On Geoscience and Remote Sensing*, 35(4):1007-1017 (November 97).
97. Wilson, Terry, et al. "Perceptual-Based Hyperspectral Image Fusion Using Multiresolution Analysis," *Optical Engineering*, 34(11):3154-3164 (November 95).

98. Younis, K., et al. "Vector Quantization Based on Dynamic Adjustment of Mahalanobis Distance." *National Aerospace and Electronics Conference (NAECON)* 2. 858-862. May 1996.
99. Younis, Khalid. *Weighted Mahalanobis Distance for Hyper-ellipsoidal Clustering*. MS thesis, Air Force Institute of Technology, 1996.
100. Zheng, Yi and James Greenleaf. "The effect of Concave and Convex Weight Adjustments on Self-Organization Maps," *IEEE Transactions on Neural Networks*, 7(1):87-96 (January 1996).

Vita

Captain Terry A. Wilson was born in Chicago Heights, Illinois on 05 January 1962. He enlisted in the Air Force in 1982 and was assigned to the Munitions Maintenance Squadron, Barksdale AFB, Louisiana. From 1982 to 1987 he worked as an Electronic Systems Analyst performing both Missile and Aircraft maintenance. He was accepted into the Airman's Education and Commissioning Program (AECPP) in 1987 and was sent to the University of Florida for his undergraduate degree in Electrical Engineering. Upon Graduation in May of 1990, he received a commission from the Air Force's Officer Training School. Captain Wilson's first assignment, after being commissioned, was to the Ballistic Missile Organization, Norton AFB, California as an Acquisition and Engineering Project Manager. From 1990 to 1993 Captain Wilson was the ICBM Code Processing System (ICPS) Project Officer for the Peacekeeper Rail Garrison (PKRG) and The Rapid Evaluation and Combat Targeting (REACT) Programs. In this capacity, Capt Wilson managed the cost and scheduling of several contracts, over-saw the design, production, and fielding of the ICPS portions of the PKRG and REACT programs, and developed the initial requirement specs for the ICPS changes in the Guidance Replacement Program (GRP). In December 1994, Captain Wilson graduated from the Air Force Institute of Technology (AFIT) with a Master of Science Degree in Electrical Engineering. He entered the Ph.D. program at AFIT in 1995.

Permanent address: 5256 Cobb Dr.
Dayton, OH 45431

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188	
<small>Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.</small>				
1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE August 1998		3. REPORT TYPE AND DATES COVERED PhD Dissertation
4. TITLE AND SUBTITLE Automatic Target Cueing Of Hyperspectral Image Data				5. FUNDING NUMBERS
6. AUTHOR(S) Terry A. Wilson, Capt, USAF				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology, WPAFB OH 45433-6583				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT/DS/ENG/98-13
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Capt Thomas Rathbun AFRL/SNAS, Area B, Bldg 23 Wright-Patterson AFB, OH 45433				10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION AVAILABILITY STATEMENT Approved for Public Release; Distribution Unlimited				12b. DISTRIBUTION CODE
13. ABSTRACT (Maximum 200 words) <p>Modern imaging sensors produce vast amounts data, overwhelming human analysts. One such sensor is the Airborne Visible and Infrared Imaging Spectrometer (AVIRIS) hyperspectral sensor. The AVIRIS sensor simultaneously collects data in 224 spectral bands that range from 0.4μm to 2.5μm in approximately 10nm increments, producing 224 images, each representing a single spectral band. Autonomous systems are required that can fuse "important" spectral bands and then classify regions of interest if all of this data is to be exploited. This dissertation presents a comprehensive solution that consists of a new physiologically motivated fusion algorithm and a novel Bayes optimal self-architecting classifier that processes the outputs of the fusion algorithm. The fusion algorithm which uses a contrast sensitivity weighted wavelet-based multiresolution analysis is shown to outperform other fusion algorithms in both visual aesthetics and signal-to-noise ratios. The self-architecting classifier is a Radial Basis Function (RBF) Iterative Construction Algorithm (RICA) that is designed to autonomously determine the size of its network architecture for optimal classification performance. RICA is shown to outperform several neural network algorithms, including a fixed architecture multi-layer Perceptron (MLP), a fixed architecture RBF, and an adaptive architecture MLP. A proof is also presented demonstrating that RICA produces a network which is a minimum mean squared-error approximate to Bayes optimal discriminant functions. Finally, it is shown that this combination of image fusion and self-architecting classifier provide an excellent means to detect targets in hyperspectral sensor data.</p>				
14. SUBJECT TERMS Target Detection, Taregt Cueing, Image Processing, Radial Basis Function Networks, Artificial Neural Networks, Image Fusion, Hyperspectral				15. NUMBER OF PAGES 116
				16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	